

# First Welfare Theorem \*

Julian Parsert      Cezary Kaliszyk

December 7, 2022

## Abstract

## Contents

<b>1</b>	<b>Introducing Syntax</b>	<b>2</b>
<b>2</b>	<b>Arg Min and Arg Max sets</b>	<b>3</b>
2.1	Definitions and Lemmas by Julian Parsert . . . . .	3
<b>3</b>	<b>Preference Relations</b>	<b>4</b>
3.1	Basic Preference Relation . . . . .	4
3.1.1	Contour sets . . . . .	5
3.2	Rational Preference Relation . . . . .	5
3.3	Finite carrier . . . . .	8
3.4	Local Non-Satiation . . . . .	11
3.5	Convex preferences . . . . .	12
3.6	Real Vector Preferences . . . . .	13
3.6.1	Monotone preferences . . . . .	16
<b>4</b>	<b>Utility Functions</b>	<b>17</b>
4.1	Ordinal utility functions . . . . .	17
4.2	Utility function on Euclidean Space . . . . .	23
<b>5</b>	<b>Consumers</b>	<b>24</b>
5.1	Pre Arrow-Debreu consumption set . . . . .	24
5.2	Arrow-Debreu model consumption set . . . . .	25
<b>6</b>	<b>Pareto Ordering</b>	<b>25</b>
6.1	Budget constraint . . . . .	26
6.2	Feasibility . . . . .	26

---

\*This work is supported by the Austrian Science Fund (FWF) project P26201 and the European Research Council (ERC) grant no 714034 *SMART*.

<b>7</b>	<b>Exchange Economy</b>	<b>27</b>
7.1	Feasibility . . . . .	28
7.2	Pareto optimality . . . . .	28
7.3	Competitive Equilibrium in Exchange Economy . . . . .	28
7.4	Lemmas for final result . . . . .	28
7.5	First Welfare Theorem in Exchange Economy . . . . .	32
<b>8</b>	<b>Pre Arrow-Debreu model</b>	<b>34</b>
8.1	Feasibility . . . . .	35
8.2	Profit maximisation . . . . .	35
8.3	Competitive Equilibrium . . . . .	35
8.4	Pareto Optimality . . . . .	36
8.5	Lemmas for final result . . . . .	36
8.6	First Welfare Theorem . . . . .	41
<b>9</b>	<b>Arrow-Debreu model</b>	<b>44</b>
9.1	Feasibility . . . . .	45
9.2	Profit maximisation . . . . .	45
9.3	Competitive Equilibrium . . . . .	46
9.4	Pareto Optimality . . . . .	46
9.5	Lemmas for final result . . . . .	47
9.6	First Welfare Theorem . . . . .	52
<b>10</b>	<b>Related work</b>	<b>55</b>

# 1 Introducing Syntax

Syntax, abbreviations and type-synonyms

```
theory Syntax
  imports Main
begin
```

```
type-synonym 'a relation = ('a × 'a) set
```

```
abbreviation gen-weak-stx :: 'a ⇒ 'a relation ⇒ 'a ⇒ bool
  (- ≥[-] - [51,100,51] 60)
where
  x ≥[P] y ≡ (x, y) ∈ P
```

```
abbreviation gen-indif-stx :: 'a ⇒ 'a relation ⇒ 'a ⇒ bool
  (- ≈[-] - [51,100,51] 60)
where
  x ≈[P] y ≡ x ≥[P] y ∧ y ≥[P] x
```

**abbreviation** *gen-strc-stx* :: 'a ⇒ 'a relation ⇒ 'a ⇒ bool  
 (- >[-] - [51,100,51] 60)  
**where**  
 $x >[P] y \equiv x \succeq[P] y \wedge \neg y \succeq[P] x$

**end**

## 2 Arg Min and Arg Max sets

**theory** *Argmax*  
**imports**  
*Complex-Main*  
**begin**

### 2.1 Definitions and Lemmas by Julian Parsert

definition of argmax and argmin returning a set.

**definition** *arg-min-set* :: ('a ⇒ 'b::ord) ⇒ 'a set ⇒ 'a set  
**where**  
 $arg-min-set f S = \{x. is-arg-min f (\lambda x. x \in S) x\}$

**definition** *arg-max-set* :: ('a ⇒ 'b::ord) ⇒ 'a set ⇒ 'a set  
**where**  
 $arg-max-set f S = \{x. is-arg-max f (\lambda x. x \in S) x\}$

Useful lemmas for *arg-max-set* and *arg-min-set*.

**lemma** *no-better-in-s*:  
**assumes**  $x \in arg-max-set f S$   
**shows**  $\nexists y. y \in S \wedge (f y) > (f x)$   
**by** (*metis arg-max-set-def assms is-arg-max-def mem-Collect-eq*)

**lemma** *argmax-sol-in-s*:  
**assumes**  $x \in arg-max-set f S$   
**shows**  $x \in S$   
**by** (*metis CollectD arg-max-set-def assms is-arg-max-def*)

**lemma** *leq-all-in-sol*:  
**fixes**  $f :: 'a \Rightarrow ('b :: preorder)$   
**assumes**  $x \in arg-max-set f S$   
**shows**  $\forall y \in S. f y \geq f x \longrightarrow y \in arg-max-set f S$   
**using** *assms le-less-trans* **by** (*auto simp: arg-max-set-def is-arg-max-def*)

**lemma** *all-leq*:  
**fixes**  $f :: 'a \Rightarrow ('b :: linorder)$   
**assumes**  $x \in arg-max-set f S$   
**shows**  $\forall y \in S. f x \geq f y$   
**by** (*meson assms leI no-better-in-s*)

```

lemma all-in-argmax-equal:
  fixes  $f :: 'a \Rightarrow ('b :: linorder)$ 
  assumes  $x \in \text{arg-max-set } f \ S$ 
  shows  $\forall y \in \text{arg-max-set } f \ S. f \ x = f \ y$ 
  by (meson all-leq argmax-sol-in-s assms le-less no-better-in-s)

end

```

### 3 Preference Relations

Preferences modeled as a set of pairs

```

theory Preferences
  imports
    HOL-Analysis.Multivariate-Analysis
    Syntax
begin

```

#### 3.1 Basic Preference Relation

Basic preference relation locale with carrier and relation modeled as a set of pairs.

```

locale preference =
  fixes  $\text{carrier} :: 'a \text{ set}$ 
  fixes  $\text{relation} :: 'a \text{ relation}$ 
  assumes not-outside:  $(x,y) \in \text{relation} \implies x \in \text{carrier}$ 
  and  $(x,y) \in \text{relation} \implies y \in \text{carrier}$ 
  assumes trans-refl: preorder-on carrier relation

```

```

context preference
begin

```

```

abbreviation geq ( $- \succeq -$  [51,51] 60)
  where
     $x \succeq y \equiv x \succeq[\text{relation}] y$ 

```

```

abbreviation str-gr ( $- \succ -$  [51,51] 60)
  where
     $x \succ y \equiv x \succeq y \wedge \neg y \succeq x$ 

```

```

abbreviation indiff ( $- \approx -$  [51,51] 60)
  where
     $x \approx y \equiv x \succeq y \wedge y \succeq x$ 

```

```

lemma reflexivity: refl-on carrier relation
  using preorder-on-def trans-refl by blast

```

**lemma** *transitivity: trans relation*  
**using** *preorder-on-def trans-refl* **by** *auto*

**lemma** *indiff-trans [simp]:  $x \approx y \implies y \approx z \implies x \approx z$*   
**by** (*meson transE transitivity*)

**end**

### 3.1.1 Contour sets

**definition** *at-least-as-good* :: '*a*  $\Rightarrow$  '*a* set  $\Rightarrow$  '*a* relation  $\Rightarrow$  '*a* set  
**where**  
*at-least-as-good* *x B P* = {*y*  $\in$  *B*. *y*  $\succeq$ [*P*] *x* }

**definition** *no-better-than* :: '*a*  $\Rightarrow$  '*a* set  $\Rightarrow$  '*a* relation  $\Rightarrow$  '*a* set  
**where**  
*no-better-than* *x B P* = {*y*  $\in$  *B*. *x*  $\succeq$ [*P*] *y*}

**definition** *as-good-as* :: '*a*  $\Rightarrow$  '*a* set  $\Rightarrow$  '*a* relation  $\Rightarrow$  '*a* set  
**where**  
*as-good-as* *x B P* = {*y*  $\in$  *B*. *x*  $\approx$ [*P*] *y*}

**lemma** *at-lst-asgd-ge:*  
**assumes** *x*  $\in$  *at-least-as-good* *y B Pr*  
**shows** *x*  $\succeq$ [*Pr*] *y*  
**using** *assms at-least-as-good-def* **by** *fastforce*

**lemma** *strict-contour-is-diff:*  
{*a*  $\in$  *B*. *a*  $\succ$ [*Pr*] *y*} = *at-least-as-good* *y B Pr* - *as-good-as* *y B Pr*  
**by**(*auto simp add: at-least-as-good-def as-good-as-def*)

**lemma** *strict-countour-def [simp]:*  
(*at-least-as-good* *y B Pr*) - *as-good-as* *y B Pr* = {*x*  $\in$  *B*. *x*  $\succ$ [*Pr*] *y*}  
**by** (*simp add: as-good-as-def at-least-as-good-def strict-contour-is-diff*)

**lemma** *at-least-as-goodD [dest]:*  
**assumes** *z*  $\in$  *at-least-as-good* *y B Pr*  
**shows** *z*  $\succeq$ [*Pr*] *y*  
**using** *assms at-least-as-good-def* **by** *fastforce*

## 3.2 Rational Preference Relation

Rational preferences add totality to the basic preferences.

**locale** *rational-preference* = *preference* +  
**assumes** *total: total-on carrier relation*  
**begin**

**lemma** *compl:  $\forall x \in carrier . \forall y \in carrier . x \succeq y \vee y \succeq x$*   
**by** (*metis refl-onD reflexivity total total-on-def*)

**lemma** *strict-not-refl-weak* [*iff*]:  $x \in \text{carrier} \wedge y \in \text{carrier} \implies \neg (y \succeq x) \longleftrightarrow x \succ y$

**by** (*metis refl-onD reflexivity total total-on-def*)

**lemma** *strict-trans* [*simp*]:  $x \succ y \implies y \succ z \implies x \succ z$

**by** (*meson transE transitivity*)

**lemma** *completeD* [*dest*]:  $x \in \text{carrier} \implies y \in \text{carrier} \implies x \neq y \implies x \succeq y \vee y \succeq x$

**by** *blast*

**lemma** *pref-in-at-least-as*:

**assumes**  $x \succeq y$

**shows**  $x \in \text{at-least-as-good } y \text{ carrier relation}$

**by** (*metis (no-types, lifting) CollectI assms*

*at-least-as-good-def preference.not-outside preference-axioms*)

**lemma** *worse-in-no-better*:

**assumes**  $x \succeq y$

**shows**  $y \in \text{no-better-than } y \text{ carrier relation}$

**by** (*metis (no-types, lifting) CollectI assms no-better-than-def*

*preference-axioms preference-def strict-not-refl-weak*)

**lemma** *strict-is-neg-transitive* :

**assumes**  $x \in \text{carrier} \wedge y \in \text{carrier} \wedge z \in \text{carrier}$

**shows**  $x \succ y \implies x \succ z \vee z \succ y$

**by** (*meson assms compl transE transitivity*)

**lemma** *weak-is-transitive*:

**assumes**  $x \in \text{carrier} \wedge y \in \text{carrier} \wedge z \in \text{carrier}$

**shows**  $x \succeq y \implies y \succeq z \implies x \succeq z$

**by** (*meson transD transitivity*)

**lemma** *no-better-than-nonepty*:

**assumes**  $\text{carrier} \neq \{\}$

**shows**  $\bigwedge x. x \in \text{carrier} \implies (\text{no-better-than } x \text{ carrier relation}) \neq \{\}$

**by** (*metis (no-types, lifting) empty-iff mem-Collect-eq*

*no-better-than-def refl-onD reflexivity*)

**lemma** *no-better-subset-pref* :

**assumes**  $x \succeq y$

**shows**  $\text{no-better-than } y \text{ carrier relation} \subseteq \text{no-better-than } x \text{ carrier relation}$

**proof**

**fix**  $a$

**assume**  $a \in \text{no-better-than } y \text{ carrier relation}$

**then show**  $a \in \text{no-better-than } x \text{ carrier relation}$

**by** (*metis (no-types, lifting) assms mem-Collect-eq no-better-than-def transE transitivity*)

qed

**lemma** *no-better-thansubset-rel* :

**assumes**  $x \in \text{carrier}$  **and**  $y \in \text{carrier}$

**assumes** *no-better-than y carrier relation*  $\subseteq$  *no-better-than x carrier relation*

**shows**  $x \succeq y$

**proof** –

**have**  $\{a \in \text{carrier}. y \succeq a\} \subseteq \{a \in \text{carrier}. x \succeq a\}$

**by** (*metis (no-types) assms(3) no-better-than-def*)

**then show** *?thesis*

**by** (*metis (mono-tags, lifting) Collect-mono-iff assms(2) compl*)

qed

**lemma** *nbt-nest* :

**shows** (*no-better-than y carrier relation*  $\subseteq$  *no-better-than x carrier relation*)  $\vee$   
(*no-better-than x carrier relation*  $\subseteq$  *no-better-than y carrier relation*)

**by** (*metis (no-types, lifting) CollectD compl no-better-subset-pref no-better-than-def not-outside subsetI*)

**lemma** *at-lst-asgd-not-ge*:

**assumes**  $\text{carrier} \neq \{\}$

**assumes**  $x \in \text{carrier}$  **and**  $y \in \text{carrier}$

**assumes**  $x \notin \text{at-least-as-good } y \text{ carrier relation}$

**shows**  $\neg x \succeq y$

**by** (*metis (no-types, lifting) CollectI assms(2) assms(4) at-least-as-good-def*)

**lemma** *as-good-as-sameIff[iff]*:

$z \in \text{as-good-as } y \text{ carrier relation} \iff z \succeq y \wedge y \succeq z$

**by** (*metis (no-types, lifting) as-good-as-def mem-Collect-eq not-outside*)

**lemma** *same-at-least-as-equal*:

**assumes**  $z \approx y$

**shows** *at-least-as-good z carrier relation* =

*at-least-as-good y carrier relation* (**is**  $?az = ?ay$ )

**proof**

**have**  $z \in \text{carrier} \wedge y \in \text{carrier}$

**by** (*meson assms refl-onD2 reflexivity*)

**moreover have**  $\forall x \in \text{carrier}. x \succeq z \longrightarrow x \succeq y$

**by** (*meson assms transD transitivity*)

**ultimately show**  $?az \subseteq ?ay$

**by** (*metis at-lst-asgd-ge at-lst-asgd-not-ge equals0D not-outside subsetI*)

**next**

**have**  $z \in \text{carrier} \wedge y \in \text{carrier}$

**by** (*meson assms refl-onD2 reflexivity*)

**moreover have**  $\forall x \in \text{carrier}. x \succeq y \longrightarrow x \succeq z$

**by** (*meson assms transD transitivity*)

**ultimately show**  $?ay \subseteq ?az$

**by** (*metis at-lst-asgd-ge at-lst-asgd-not-ge*)

*equals0D not-outside subsetI*)

**qed**

**lemma** *as-good-asIff [iff]*:

*x ∈ as-good-as y carrier relation*  $\longleftrightarrow$  *x ≈[relation] y*

**by** *blast*

**lemma** *nbt-subset*:

**assumes** *finite carrier*

**assumes** *x ∈ carrier and y ∈ carrier*

**shows** *no-better-than x carrier relation*  $\subseteq$  *no-better-than x carrier relation*  $\vee$   
*no-better-than x carrier relation*  $\subseteq$  *no-better-than x carrier relation*

**by** *auto*

**lemma** *fnt-carrier-fnt-rel: finite carrier  $\implies$  finite relation*

**by** (*metis finite-SigmaI refl-on-def reflexivity rev-finite-subset*)

**lemma** *nbt-subset-carrier*:

**assumes** *x ∈ carrier*

**shows** *no-better-than x carrier relation*  $\subseteq$  *carrier*

**using** *no-better-than-def* **by** *fastforce*

**lemma** *xy-in-eachothers-nbt*:

**assumes** *x ∈ carrier y ∈ carrier*

**shows** *x ∈ no-better-than y carrier relation*  $\vee$   
*y ∈ no-better-than x carrier relation*

**by** (*meson assms(1) assms(2) contra-subsetD nbt-nest refl-onD reflexivity worse-in-no-better*)

**lemma** *same-nbt-same-pref*:

**assumes** *x ∈ carrier y ∈ carrier*

**shows** *x ∈ no-better-than y carrier relation*  $\wedge$

*y ∈ no-better-than x carrier relation*  $\longleftrightarrow$  *x ≈ y*

**by** (*metis (mono-tags, lifting) CollectD contra-subsetD no-better-subset-pref*  
*no-better-than-def worse-in-no-better*)

**lemma** *indifferent-imp-weak-pref*:

**assumes** *x ≈ y*

**shows** *x  $\succeq$  y y  $\succeq$  x*

**by** (*simp add: assms*)**+**

### 3.3 Finite carrier

**context**

**assumes** *finite carrier*

**begin**

**lemma** *fnt-carrier-fnt-nbt*:

**shows**  $\forall x \in \text{carrier. finite (no-better-than x carrier relation)}$

**proof**



**fix**  $x$   
**assume**  $x \in \text{carrier}$   
**then show**  $\text{finite (no-better-than } x \text{ carrier relation)}$   
**using**  $\text{finite-subset nbt-subset-carrier } \langle \text{finite carrier} \rangle$  **by**  $\text{blast}$   
**qed**

**lemma**  $\text{nbt-subset-imp-card-leq}$ :

**assumes**  $x \in \text{carrier}$  **and**  $y \in \text{carrier}$   
**shows**  $\text{no-better-than } x \text{ carrier relation} \subseteq \text{no-better-than } y \text{ carrier relation} \longleftrightarrow$   
 $\text{card (no-better-than } x \text{ carrier relation)} \leq \text{card (no-better-than } y \text{ carrier relation)}$   
**(is**  $?nbt \longleftrightarrow ?card$ **)**

**proof**

**assume**  $?nbt$   
**then show**  $?card$   
**by**  $(\text{simp add: assms(2) card-mono fnt-carrier-fnt-nbt})$

**next**

**assume**  $?card$   
**then show**  $?nbt$   
**by**  $(\text{metis assms(1) card-seteq fnt-carrier-fnt-nbt nbt-nest})$

**qed**

**lemma**  $\text{card-leq-pref}$ :

**assumes**  $x \in \text{carrier}$  **and**  $y \in \text{carrier}$   
**shows**  $\text{card (no-better-than } x \text{ carrier relation)} \leq \text{card (no-better-than } y \text{ carrier relation)}$   
 $\longleftrightarrow y \succeq x$

**proof**  $(\text{rule iffI, goal-cases})$

**case** 1

**then show**  $?case$   
**using**  $\text{assms(1) assms(2) nbt-subset-imp-card-leq no-better-than-subset-rel}$  **by**  $\text{presburger}$

**next**

**case** 2

**then show**  $?case$   
**using**  $\text{assms(1) assms(2) nbt-subset-imp-card-leq no-better-subset-pref}$  **by**  $\text{blast}$

**qed**

**lemma**  $\text{finite-ne-remove-induct}$ :

**assumes**  $\text{finite } B \ B \neq \{\}$   
 $\bigwedge A. \text{finite } A \implies A \subseteq B \implies A \neq \{\} \implies$   
 $(\bigwedge x. x \in A \implies A - \{x\} \neq \{\} \implies P (A - \{x\})) \implies P A$   
**shows**  $P B$   
**by**  $(\text{metis assms finite-remove-induct}[\text{where } P = \lambda F. F = \{\} \vee P F \text{ for } P])$

**lemma**  $\text{finite-nempty-preorder-has-max}$ :

**assumes**  $\text{finite } B \ B \neq \{\}$   $\text{refl-on } B \ R$   $\text{trans } R$   $\text{total-on } B \ R$   
**shows**  $\exists x \in B. \forall y \in B. (x, y) \in R$   
**using**  $\text{assms(1) subset-refl[of } B] \text{ assms(2)}$

**proof** (*induct B rule: finite-subset-induct*)  
**case** (*insert x F*)  
**then show** *?case using assms(3-)*  
**by** (*cases F = {}*) (*auto simp: refl-onD total-on-def, metis refl-onD2 transE*)  
**qed** *auto*

**lemma** *finite-nempty-preorder-has-min:*  
**assumes** *finite B B ≠ {} refl-on B R trans R total-on B R*  
**shows**  $\exists x \in B. \forall y \in B. (y, x) \in R$   
**using** *assms(1) subset-refl[of B] assms(2)*  
**proof** (*induct B rule: finite-subset-induct*)  
**case** (*insert x F*)  
**then show** *?case using assms(3-)*  
**by** (*cases F = {}*) (*auto simp: refl-onD total-on-def, metis refl-onD2 transE*)  
**qed** *auto*

**lemma** *finite-nonempty-carrier-has-maximum:*  
**assumes** *carrier ≠ {}*  
**shows**  $\exists e \in \text{carrier}. \forall m \in \text{carrier}. e \succeq[\text{relation}] m$   
**using** *finite-nempty-preorder-has-max[of carrier relation] assms*  
*⟨finite carrier⟩ reflexivity total transitivity by blast*

**lemma** *finite-nonempty-carrier-has-minimum:*  
**assumes** *carrier ≠ {}*  
**shows**  $\exists e \in \text{carrier}. \forall m \in \text{carrier}. m \succeq[\text{relation}] e$   
**using** *finite-nempty-preorder-has-min[of carrier relation] assms*  
*⟨finite carrier⟩ reflexivity total transitivity by blast*

**end**

**lemma** *all-carrier-ex-sub-rel:*  
 $\forall c \subseteq \text{carrier}. \exists r \subseteq \text{relation}. \text{rational-preference } c \ r$   
**proof** (*standard,standard*)  
**fix** *c*  
**assume** *c-in: c ⊆ carrier*  
**define** *r' where*  
 $r' = \{(x,y) \in \text{relation}. x \in c \wedge y \in c\}$   
**have** *r'-sub: r' ⊆ c × c*  
**using** *r'-def by blast*  
**have**  $\forall x \in c. x \succeq[r'] x$   
**by** (*metis (no-types, lifting) CollectI c-in case-prodI compl r'-def subsetCE*)  
**then have** *refl: refl-on c r'*  
**by** (*meson r'-sub refl-onI*)  
**have** *trans: trans r'*  
**proof**  
**fix** *x y z*  
**assume** *a1: x ⪰[r'] y*  
**assume** *a2: y ⪰[r'] z*

```

show  $x \succeq[r'] z$ 
by (metis (mono-tags, lifting) CollectD CollectI a1 a2 case-prodD case-prodI
r'-def transE transitivity)
qed
have total: total-on c r'
proof (standard)
  fix  $x y$ 
  assume  $a1: x \in c$ 
  assume  $a2: y \in c$ 
  assume  $a3: x \neq y$ 
  show  $x \succeq[r'] y \vee y \succeq[r'] x$ 
  by (metis (no-types, lifting) CollectI a1 a2 c-in case-prodI compl r'-def sub-
set-iff)
qed
have rational-preference c r'
by (meson local.refl local.trans preference.intro preorder-on-def rational-preference.intro

rational-preference-axioms.intro refl-on-domain total)
thus  $\exists r \subseteq \text{relation. rational-preference } c r$ 
by (metis (no-types, lifting) CollectD case-prodD r'-def subrelI)
qed

end

```

### 3.4 Local Non-Satiation

Defining local non-satiation.

**definition** *local-nonsatiation*

**where**

*local-nonsatiation*  $B P \longleftrightarrow$

$(\forall x \in B. \forall e > 0. \exists y \in B. \text{norm } (y - x) \leq e \wedge y \succ[P] x)$

Alternate definitions and intro/dest rules with them

**lemma** *lns-alt-def1 [iff]* :

**shows** *local-nonsatiation*  $B P \longleftrightarrow (\forall x \in B. \forall e > 0. (\exists y \in B. \text{dist } y x \leq e \wedge y \succ[P] x))$

**by** (*simp add : dist-norm local-nonsatiation-def*)

**lemma** *lns-normI [intro]*:

**assumes**  $\bigwedge x e. x \in B \implies e > 0 \implies (\exists y \in B. \text{norm } (y - x) \leq e \wedge y \succ[P] x)$

**shows** *local-nonsatiation*  $B P$

**by** (*simp add: assms dist-norm*)

**lemma** *lns-distI [intro]*:

**assumes**  $\bigwedge x e. x \in B \implies e > 0 \implies (\exists y \in B. (\text{dist } y x) \leq e \wedge y \succ[P] x)$

**shows** *local-nonsatiation*  $B P$

**by** (*meson lns-alt-def1 assms*)

**lemma** *lns-alt-def2 [iff]*:

*local-nonsatiation*  $B P \longleftrightarrow (\forall x \in B. \forall e > 0. (\exists y. y \in (\text{ball } x e) \wedge y \in B \wedge y \succ [P] x))$

**proof**

**assume** *local-nonsatiation*  $B P$

**then show**  $\forall x \in B. \forall e > 0. \exists x'. x' \in \text{ball } x e \wedge x' \in B \wedge x' \succ [P] x$

**by** (*auto simp add : ball-def*) (*metis dense le-less-trans dist-commute*)

**next**

**assume**  $\forall x \in B. \forall e > 0. \exists y. y \in \text{ball } x e \wedge y \in B \wedge y \succ [P] x$

**then show** *local-nonsatiation*  $B P$

**by** (*metis (no-types, lifting) ball-def dist-commute*

*less-le-not-le lns-alt-def1 mem-Collect-eq*)

**qed**

**lemma** *lns-normD* [*dest*]:

**assumes** *local-nonsatiation*  $B P$

**shows**  $\forall x \in B. \forall e > 0. \exists y \in B. (\text{norm } (y - x) \leq e \wedge y \succ [P] x)$

**by** (*meson assms local-nonsatiation-def*)

### 3.5 Convex preferences

**definition** *weak-convex-pref* :: (*'a::real-vector*) *relation*  $\Rightarrow$  *bool*

**where**

*weak-convex-pref*  $Pr \longleftrightarrow (\forall x y. x \succeq [Pr] y \longrightarrow$

$(\forall \alpha \beta. \alpha + \beta = 1 \wedge \alpha > 0 \wedge \beta > 0 \longrightarrow \alpha *_R x + \beta *_R y \succeq [Pr] y))$

**definition** *convex-pref* :: (*'a::real-vector*) *relation*  $\Rightarrow$  *bool*

**where**

*convex-pref*  $Pr \longleftrightarrow (\forall x y. x \succ [Pr] y \longrightarrow$

$(\forall \alpha. 1 > \alpha \wedge \alpha > 0 \longrightarrow \alpha *_R x + (1 - \alpha) *_R y \succ [Pr] y))$

**definition** *strict-convex-pref* :: (*'a::real-vector*) *relation*  $\Rightarrow$  *bool*

**where**

*strict-convex-pref*  $Pr \longleftrightarrow (\forall x y. x \succeq [Pr] y \wedge x \neq y \longrightarrow$

$(\forall \alpha. 1 > \alpha \wedge \alpha > 0 \longrightarrow \alpha *_R x + (1 - \alpha) *_R y \succ [Pr] y))$

**lemma** *convex-ge-imp-conved*:

**assumes**  $\forall x y. x \succeq [Pr] y \longrightarrow (\forall \alpha \beta. \alpha + \beta = 1 \wedge \alpha \geq 0 \wedge \beta \geq 0 \longrightarrow \alpha *_R x + \beta *_R y \succeq [Pr] y)$

**shows** *weak-convex-pref*  $Pr$

**by** (*simp add: assms weak-convex-pref-def*)

**lemma** *weak-convexI* [*intro*]:

**assumes**  $\bigwedge x y \alpha \beta. x \succeq [Pr] y \implies \alpha + \beta = 1 \implies 0 < \alpha \implies 0 < \beta \implies \alpha *_R x + \beta *_R y \succeq [Pr] y$

**shows** *weak-convex-pref*  $Pr$

**by** (*simp add: assms weak-convex-pref-def*)

**lemma** *weak-convexD* [*dest*]:

**assumes** *weak-convex-pref*  $Pr$  **and**  $x \succeq [Pr] y$  **and**  $0 < u$  **and**  $0 < v$  **and**  $u +$

$v = 1$   
**shows**  $u *_R x + v *_R y \succeq[Pr] y$   
**using** *assms weak-convex-pref-def* **by** *blast*

### 3.6 Real Vector Preferences

Preference relations on real vector type class.

**locale** *real-vector-rpr = rational-preference carrier relation*  
**for** *carrier :: 'a::real-vector set*  
**and** *relation :: 'a relation*

**sublocale** *real-vector-rpr  $\subseteq$  rational-preference carrier relation*  
**by** (*simp add: rational-preference-axioms*)

**context** *real-vector-rpr*  
**begin**

**lemma** *have-rpr: rational-preference carrier relation*  
**by** (*simp add: rational-preference-axioms*)

Multiple convexity alternate definitions intro/dest rules.

**lemma** *weak-convex1D [dest]:*

**assumes** *weak-convex-pref relation* **and**  $x \succeq[relation] y$  **and**  $0 \leq u$  **and**  $0 \leq v$   
**and**  $u + v = 1$

**shows**  $u *_R x + v *_R y \succeq[relation] y$

**proof** –

**have**  $u-0: u = 0 \longrightarrow u *_R x + v *_R y \succeq[relation] y$

**proof**

**assume**  $u-0: u = 0$

**have**  $v = 1$

**using** *assms(5) u-0* **by** *auto*

**then have** *?thesis*

**by** (*metis add.left-neutral assms(2) preference.reflexivity preference-axioms  
real-vector.scale-zero-left refl-onD2 scaleR-one strict-not-refl-weak*)

**thus**  $u *_R x + v *_R y \succeq[relation] y$

**using**  $u-0$  **by** *blast*

**qed**

**have**  $u \neq 0 \wedge u \neq 1 \longrightarrow u *_R x + v *_R y \succeq[relation] y$

**by** (*metis add-cancel-right-right antisym-conv not-le assms weak-convexD*)

**then show** *?thesis*

**by** (*metis u-0 assms(2,5) add-cancel-right-right real-vector.scale-zero-left scaleR-one*)

**qed**

**lemma** *weak-convex1I [intro] :*

**assumes**  $\forall x. convex$  (*at-least-as-good x carrier relation*)

**shows** *weak-convex-pref relation*

**proof** (*rule weak-convexI*)

**fix**  $x$  **and**  $y$  **and**  $\alpha \beta :: real$

**assume** *assum :  $x \succeq[relation] y$*

```

assume reals:  $0 < \alpha$   $0 < \beta$   $\alpha + \beta = 1$ 
then have  $x \in \text{carrier}$ 
  by (meson assum preference.not-outside rational-preference.axioms(1) have-rpr)
have  $y \in \text{carrier}$ 
  by (meson assum refl-onD2 reflexivity)
then have  $y \text{-in-upper-cont}$ :  $y \in (\text{at-least-as-good } y \text{ carrier relation})$ 
  using assms rational-preference.at-lst-asgd-not-ge
    rational-preference.compl by (metis empty-iff have-rpr)
then have  $x \in (\text{at-least-as-good } y \text{ carrier relation})$ 
  using assum pref-in-at-least-as by blast
moreover have  $0 \leq \beta$  and  $0 \leq \alpha$ 
  using reals by (auto)
ultimately show  $(\alpha *_R x + \beta *_R y) \succeq[\text{relation}] y$ 
  by (meson assms(1) at-least-as-goodD convexD reals(3) y-in-upper-cont)
qed

```

Definition of convexity in "Handbook of Social Choice and Welfare"[1].

**lemma** *convex-def-alt*:

```

assumes rational-preference carrier relation
assumes weak-convex-pref relation
shows  $(\forall x \in \text{carrier}. \text{convex } (\text{at-least-as-good } x \text{ carrier relation}))$ 
proof
  fix  $x$ 
  assume  $x \text{-in}$ :  $x \in \text{carrier}$ 
  show  $\text{convex } (\text{at-least-as-good } x \text{ carrier relation})$  (is convex ?x)
  proof (rule convexI)
    fix  $a$   $b$  :: ' $a$  and  $\alpha$  :: real and  $\beta$  :: real
    assume  $a \text{-in}$ :  $a \in ?x$ 
    assume  $b \text{-in}$ :  $b \in ?x$ 
    assume reals:  $0 \leq \alpha$   $0 \leq \beta$   $\alpha + \beta = 1$ 
    have  $a \text{-g-}x$ :  $a \succeq[\text{relation}] x$ 
      using  $a \text{-in}$  by blast
    have  $b \text{-g-}x$ :  $b \succeq[\text{relation}] x$ 
      using  $b \text{-in}$  by blast
    have  $a \succeq[\text{relation}] b \vee b \succeq[\text{relation}] a$ 
      by (meson a-in at-least-as-goodD b-in preference.not-outside
        rational-preference.compl rational-preference-def assms(1))
    then show  $\alpha *_R a + \beta *_R b \in ?x$ 
    proof (rule disjE)
      assume  $a \succeq[\text{relation}] b$ 
      then have  $\alpha *_R a + \beta *_R b \succeq[\text{relation}] b$ 
        using assms reals by blast
      then have  $\alpha *_R a + \beta *_R b \succeq[\text{relation}] x$ 
        by (meson b-g-x assms(1) preference.not-outside x-in
          rational-preference.strict-is-neg-transitive
          rational-preference.strict-not-refl-weak rational-preference-def)
      then show ?thesis
        by (metis (no-types, lifting) CollectI assms(1)
          at-least-as-good-def preference-def rational-preference-def)

```

```

next
  assume as:  $b \succeq[\text{relation}] a$ 
  then have  $\alpha *_R a + \beta *_R b \succeq[\text{relation}] a$ 
    by (metis add commute assms(2) reals weak-convex1D)
  have  $\alpha *_R a + \beta *_R b \succeq[\text{relation}] a$ 
    by (metis as add commute assms(2)
        reals(1,2,3) weak-convex1D)
  then have  $\alpha *_R a + \beta *_R b \succeq[\text{relation}] x$ 
    by (meson a-g-x assms(1) preference.indiff-trans x-in
        preference.not-outside rational-preference.axioms(1)
        rational-preference.strict-is-neg-transitive )
  then show ?thesis
    using pref-in-at-least-as by blast
qed
qed
qed

lemma convex-imp-convex-str-upper-cnt:
  assumes  $\forall x \in \text{carrier}. \text{convex} (\text{at-least-as-good } x \text{ carrier relation})$ 
  shows  $\text{convex} (\text{at-least-as-good } x \text{ carrier relation} - \text{as-good-as } x \text{ carrier relation})$ 
    (is  $\text{convex} ( ?a - ?b)$ )
proof (rule convexI)
  fix a y u v
  assume as-a:  $a \in ?a - ?b$ 
  assume as-y:  $y \in ?a - ?b$ 
  assume reals:  $0 \leq (u::\text{real}) \ 0 \leq v \ u + v = 1$ 
  have cvx: weak-convex-pref relation
    by (meson assms at-least-as-goodD convexI have-rpr
        preference-def rational-preference.axioms(1) weak-convex1I)
  then have a-g-x:  $a \succ[\text{relation}] x$ 
    using as-a by blast
  then have y-gt-x:  $y \succ[\text{relation}] x$ 
    using as-y by blast
  show  $u *_R a + v *_R y \in ?a - ?b$ 
proof
  show  $u *_R a + v *_R y \in ?a$ 
    by (metis DiffD1 a-g-x as-a as-y assms convexD reals have-rpr
        preference-def rational-preference.axioms(1))
next
  have  $a \succeq[\text{relation}] y \vee y \succeq[\text{relation}] a$ 
    by (meson a-g-x y-gt-x assms(1) preference.not-outside have-rpr
        rational-preference.axioms(1) rational-preference.strict-not-refl-weak)
  then show  $u *_R a + v *_R y \notin ?b$ 
proof
  assume  $a \succeq[\text{relation}] y$ 
  then have  $u *_R a + v *_R y \succeq[\text{relation}] y$ 
    using cvx assms(1) reals by blast
  then have  $u *_R a + v *_R y \succ[\text{relation}] x$ 
    using y-gt-x by (meson assms(1) rational-preference.axioms(1) have-rpr

```

```

      rational-preference.strict-is-neg-transitive preference-def)
    then show  $u *_R a + v *_R y \notin \text{as-good-as } x \text{ carrier relation}$ 
      by blast
  next
  assume  $y \succeq[\text{relation}] a$ 
  then have  $u *_R a + v *_R y \succeq[\text{relation}] a$ 
    using cvx assms(1) reals by (metis add.commute weak-convex1D)
  then have  $u *_R a + v *_R y \succ[\text{relation}] x$ 
    by (meson a-g-x assms(1) rational-preference.strict-is-neg-transitive
      rational-preference.axioms(1) preference-def have-rpr)
  then show  $u *_R a + v *_R y \notin ?b$ 
    by blast
  qed
qed
qed
end

```

### 3.6.1 Monotone preferences

**definition** *weak-monotone-prefs* :: 'a set  $\Rightarrow$  ('a::ord) relation  $\Rightarrow$  bool  
 where  
 $\text{weak-monotone-prefs } B P \longleftrightarrow (\forall x \in B. \forall y \in B. x \geq y \longrightarrow x \succeq[P] y)$

**definition** *monotone-preference* :: 'a set  $\Rightarrow$  ('a::ord) relation  $\Rightarrow$  bool  
 where  
 $\text{monotone-preference } B P \longleftrightarrow (\forall x \in B. \forall y \in B. x > y \longrightarrow x \succ[P] y)$

Given a carrier set that is unbounded above (not the "standard" mathematical definition), monotonicity implies local non-satiation.

**lemma** *unbounded-above-mono-imp-lns*:  
 assumes  $\forall M \in \text{carrier}. (\forall x > M. x \in \text{carrier})$   
 assumes *mono*: *monotone-preference* (*carrier*:: 'a::ordered-euclidean-space set)  
 relation

shows *local-nonsatiation carrier relation*  
**proof**(*rule lns-dist1*)  
 fix  $x$  and  $e$ ::real  
 assume *x-in*:  $x \in \text{carrier}$   
 assume *gz* :  $e > 0$   
 show  $\exists y \in \text{carrier}. \text{dist } y x \leq e \wedge y \succeq[\text{relation}] x \wedge (x, y) \notin \text{relation}$   
**proof** –  
 obtain  $v$  :: real where  
 $v : v < e \wedge 0 < v$  using *gz dense* by blast  
 obtain  $i$  where  
 $i : (i :: 'a) \in \text{Basis}$  by *fastforce*  
 define  $y$  where  
 $y\text{-value} : y = x + v *_R i$   
 have *ge*:  $y \geq x$   
 using *y-value i unfolding y-value*



```

    by (simp add: v(2) zero-le-scaleR-iff)
  have  $y \neq x$ 
    using y-value i unfolding y-value
    using v(2) by auto
  hence y-str-g-x : y > x
    using ge by auto
  have y-in: y ∈ carrier
    using assms(1) x-in y-str-g-x by blast
  then have y-pref-x : y ≻[relation] x
    using y-str-g-x x-in mono monotone-preference-def by blast
  hence norm (y - x) ≤ e
    using  $\langle 0 < v \rangle$  y-value y-value i v by auto
  hence dist-less-e : dist y x ≤ e
    by (simp add: dist-norm)
  thus ?thesis
    using y-pref-x dist-less-e y-in by blast
qed
qed
end

```

## 4 Utility Functions

Utility functions and results involving them.

```

theory Utility-Functions
  imports
    Preferences
begin

```

### 4.1 Ordinal utility functions

Ordinal utility function locale

```

locale ordinal-utility =
  fixes carrier :: 'a set
  fixes relation :: 'a relation
  fixes u :: 'a ⇒ real
  assumes util-def[iff]: x ∈ carrier ⇒ y ∈ carrier ⇒ x ≻[relation] y ⟷ u x
     $\geq u y$ 
  assumes not-outside: x ≻[relation] y ⇒ x ∈ carrier
    and  $x \succeq[relation] y \implies y \in carrier$ 
begin

```

```

lemma util-def-conf: x ∈ carrier ⇒ y ∈ carrier ⇒ u x ≥ u y ⟷ x ≻[relation]
  y
  using util-def by blast

```

```

lemma relation-subset-crossp:

```

$relation \subseteq carrier \times carrier$   
**proof**  
**fix**  $x$   
**assume**  $x \in relation$   
**have**  $\forall (a,b) \in relation. a \in carrier \wedge b \in carrier$   
**by** (*metis (no-types, lifting) case-prod-conv ordinal-utility-axioms ordinal-utility-def surj-pair*)  
**then show**  $x \in carrier \times carrier$   
**using**  $\langle x \in relation \rangle$  **by** *auto*  
**qed**

Utility function implies totality of relation

**lemma** *util-imp-total: total-on carrier relation*

**proof**  
**fix**  $x$  **and**  $y$   
**assume**  $x-inc: x \in carrier$  **and**  $y-inc: y \in carrier$   
**have**  $fst : u x \geq u y \vee u y \geq u x$   
**using** *util-def* **by** *auto*  
**then show**  $x \succeq[relation] y \vee y \succeq[relation] x$   
**by** (*simp add: x-inc y-inc*)  
**qed**

**lemma** *x-y-in-carrier:  $x \succeq[relation] y \implies x \in carrier \wedge y \in carrier$*

**by** (*meson ordinal-utility-axioms ordinal-utility-def*)

Utility function implies transitivity of relation.

**lemma** *util-imp-trans: trans relation*

**proof** (*rule transI*)  
**fix**  $x$  **and**  $y$  **and**  $z$   
**assume**  $x-y: x \succeq[relation] y$   
**assume**  $y-z: y \succeq[relation] z$   
**have**  $x-ge-y: x \succeq[relation] y$   
**using**  $x-y$  **by** *auto*  
**then have**  $x-y: u x \geq u y$   
**by** (*meson x-y-in-carrier ordinal-utility-axioms util-def x-y*)  
**have**  $u y \geq u z$   
**by** (*meson y-z ordinal-utility-axioms ordinal-utility-def*)  
**have**  $x \in carrier$   
**using**  $x-y-in-carrier[of x y] x-ge-y$  **by** *simp*  
**then have**  $u x \geq u z$   
**using**  $\langle u z \leq u y \rangle$  *order-trans x-y* **by** *blast*  
**hence**  $x \succeq[relation] z$   
**by** (*meson  $\langle x \in carrier \rangle$  ordinal-utility-axioms ordinal-utility-def y-z*)  
**then show**  $x \succeq[relation] z$  .  
**qed**

**lemma** *util-imp-refl: refl-on carrier relation*

**by** (*simp add: refl-on-def relation-subset-crossp*)

```

lemma affine-trans-is-u:
  shows  $\forall \alpha > 0. (\forall \beta. \text{ordinal-utility carrier relation } (\lambda x. u(x) * \alpha + \beta))$ 
proof (rule allI, rule impI, rule allI)
  fix  $\alpha :: \text{real}$  and  $\beta$ 
  assume  $*: \alpha > 0$ 
  show ordinal-utility carrier relation  $(\lambda x. u x * \alpha + \beta)$ 
  proof (subst ordinal-utility-def, rule conjI, goal-cases)
    case 1
    then show ?case
      by (metis * add commute add-le-cancel-left not-le mult-less-iff1 util-def-conf)
    next
    case 2
    then show ?case
      by (meson refl-on-domain util-imp-refl)
  qed
qed

```

This utility function definition is ordinal. Hence they are only unique up to a monotone transformation.

```

lemma ordinality-of-utility-function :
  fixes  $f :: \text{real} \Rightarrow \text{real}$ 
  assumes monot: monotone ( $>$ ) ( $>$ )  $f$ 
  shows  $(f \circ u) x > (f \circ u) y \longleftrightarrow u x > u y$ 
proof –
  let ?func =  $(\lambda x. f(u x))$ 
  have  $\forall m n . u m \geq u n \longleftrightarrow ?func m \geq ?func n$ 
    by (metis le-less monot monotone-def not-less)
  hence  $u x > u y \longleftrightarrow ?func x > ?func y$ 
    using not-le by blast
  thus ?thesis by auto
qed

```

```

corollary utility-prefs-corresp :
  fixes  $f :: \text{real} \Rightarrow \text{real}$ 
  assumes monotonicity : monotone ( $>$ ) ( $>$ )  $f$ 
  shows  $\forall x \in \text{carrier}. \forall y \in \text{carrier}. (x,y) \in \text{relation} \longleftrightarrow (f \circ u) x \geq (f \circ u) y$ 
  by (meson monotonicity not-less ordinality-of-utility-function util-def-conf)

```

```

corollary monotone-comp-is-utility:
  fixes  $f :: \text{real} \Rightarrow \text{real}$ 
  assumes monot: monotone ( $>$ ) ( $>$ )  $f$ 
  shows ordinal-utility carrier relation  $(f \circ u)$ 
proof (rule ordinal-utility.intro, goal-cases)
case (1  $x y$ )
  then show ?case
    using monot utility-prefs-corresp by blast
next
  case (2  $x y$ )
  then show ?case

```

```

    using not-outside by blast
next
case ( $\exists x y$ )
then show ?case
    using x-y-in-carrier by blast
qed

```

```

lemma ordinal-utility-left:
  assumes  $x \succeq[\textit{relation}] y$ 
  shows  $u x \geq u y$ 
  using assms x-y-in-carrier by blast

```

```

lemma add-right:
  assumes  $\bigwedge x y. x \succeq[\textit{relation}] y \implies f x \geq f y$ 
  shows ordinal-utility carrier relation ( $\lambda x. u x + f x$ )
proof (rule ordinal-utility.intro, goal-cases)
  case ( $1 x y$ )
  assume xy:  $x \in \textit{carrier} \ y \in \textit{carrier}$ 
  then show ?case
  proof -
    have  $u x \leq u y \longrightarrow (\exists r. ((x, y) \notin \textit{relation} \wedge \neg r \leq u x + f x) \wedge r \leq u y + f y) \vee u y \leq u x$ 
    by (metis (no-types) add-le-cancel-left add-le-cancel-right assms util-def xy(1) xy(2))
    moreover show ?thesis
    by (meson add-mono assms calculation le-cases order-trans util-def xy(1) xy(2))
  qed
next
case ( $2 x y$ )
then show ?case
    using not-outside by blast
next
case ( $3 x y$ )
then show ?case
    using x-y-in-carrier by blast
qed

```

```

lemma add-left:
  assumes  $\bigwedge x y. x \succeq[\textit{relation}] y \implies f x \geq f y$ 
  shows ordinal-utility carrier relation ( $\lambda x. f x + u x$ )
proof -
  have ordinal-utility carrier relation ( $\lambda x. u x + f x$ )
  by (simp add: add-right assms)
  thus ?thesis using Groups.ab-semigroup-add-class.add commute
  by (simp add: add commute)
qed

```

```

lemma ordinal-utility-scale-transl:
  assumes (c::real) > 0
  shows ordinal-utility carrier relation ( $\lambda x. c * (u x) + d$ )
proof –
  have monotone (>) (>) ( $\lambda x. c * x + d$ ) (is monotone (>) (>) ?fn )
    by (simp add: assms monotone-def)
  with monotone-comp-is-utility have ordinal-utility carrier relation (?fn o u)
    by blast
  moreover have ?fn o u = ( $\lambda x. c * (u x) + d$ )
    by auto
  finally show ?thesis
    by auto
qed

```

```

lemma strict-preference-iff-strict-utility:
  assumes x ∈ carrier
  assumes y ∈ carrier
  shows x  $\succ$ [relation] y  $\longleftrightarrow$  u x > u y
  by (meson assms(1) assms(2) less-eq-real-def not-less-util-def)

```

**end**

A utility function implies a rational preference relation. Hence a utility function contains exactly the same amount of information as a RPR

```

sublocale ordinal-utility  $\subseteq$  rational-preference carrier relation

```

**proof**

```

  fix x and y
  assume xy: x  $\succeq$ [relation] y
  then show x ∈ carrier
    and y ∈ carrier
    using not-outside by (simp)
    (meson xy refl-onD2 util-imp-refl)
next
  show preorder-on carrier relation
  proof –
    have trans relation using util-imp-trans by auto
    then have preorder-on carrier relation
      by (simp add: preorder-on-def util-imp-refl)
    then show ?thesis .
  qed
next
  show total-on carrier relation
    by (simp add: util-imp-total)
qed

```

Given a finite carrier set. We can guarantee that given a rational preference relation, there must also exist a utility function representing this relation. Construction of witness roughly follows from.

```

theorem fnt-carrier-exists-util-fun:

```

```

assumes finite carrier
assumes rational-preference carrier relation
shows  $\exists u.$  ordinal-utility carrier relation u
proof –
  define f where
    f: f = ( $\lambda x.$  card (no-better-than x carrier relation))
  have ordinal-utility carrier relation f
  proof
    fix x y
    assume x-c:  $x \in \text{carrier}$ 
    assume y-c:  $y \in \text{carrier}$ 
    show  $x \succeq[\text{relation}] y \iff (\text{real } (f y) \leq \text{real } (f x))$ 
    proof
      assume asm:  $x \succeq[\text{relation}] y$ 
      define yn where
        yn: yn = no-better-than y carrier relation
      define xn where
        xn: xn = no-better-than x carrier relation
      then have  $yn \subseteq xn$ 
      by (simp add: asm yn assms(2) rational-preference.no-better-subset-pref)
      then have  $\text{card } yn \leq \text{card } xn$ 
      by (simp add: x-c y-c asm assms(1) assms(2) rational-preference.card-leq-pref
xn yn)
      then show  $(\text{real } (f y) \leq \text{real } (f x))$ 
      using f xn yn by simp
    next
      assume  $\text{real } (f y) \leq \text{real } (f x)$ 
      then show  $x \succeq[\text{relation}] y$ 
      using assms(1) assms(2) f rational-preference.card-leq-pref x-c y-c by
fastforce
      qed
    next
      fix x y
      assume asm:  $x \succeq[\text{relation}] y$ 
      show  $x \in \text{carrier}$ 
      by (meson asm assms(2) preference.not-outside rational-preference.axioms(1))
      show  $y \in \text{carrier}$ 
      by (meson asm assms(2) preference-def rational-preference-def)
      qed
    then show ?thesis
    by blast
  qed

```

```

corollary obt-u-fnt-carrier:
  assumes finite carrier
  assumes rational-preference carrier relation
  obtains u where ordinal-utility carrier relation u
  using assms(1) assms(2) fnt-carrier-exists-util-fun by blast

```

**theorem** *ordinal-util-imp-rat-prefs*:  
**assumes** *ordinal-utility carrier relation u*  
**shows** *rational-preference carrier relation*  
**by** (*metis (full-types) assms order-on-defs(1) ordinal-utility.util-imp-refl*  
*ordinal-utility.util-imp-total ordinal-utility.util-imp-trans ordinal-utility-def*  
*preference.intro rational-preference.intro rational-preference-axioms-def*)

## 4.2 Utility function on Euclidean Space

**locale** *eucl-ordinal-utility = ordinal-utility carrier relation u*  
**for** *carrier :: ('a::euclidean-space) set*  
**and** *relation :: 'a relation*  
**and** *u :: 'a  $\Rightarrow$  real*

**sublocale** *eucl-ordinal-utility  $\subseteq$  rational-preference carrier relation*  
**using** *rational-preference-axioms by blast*

**lemma** *ord-eucl-utility-imp-rpr: eucl-ordinal-utility s rel u  $\longrightarrow$  real-vector-rpr s rel*  
**using** *eucl-ordinal-utility.axioms ordinal-util-imp-rat-prefs real-vector-rpr.intro*  
**by** *blast*

**context** *eucl-ordinal-utility*  
**begin**

Local non-satiation on utility functions

**lemma** *lns-pref-lns-util [iff]*:  
*local-nonsatiation carrier relation  $\longleftrightarrow$*   
*( $\forall x \in \text{carrier}. \forall e > 0. \exists y \in \text{carrier}.$*   
 *$\text{norm } (y - x) \leq e \wedge u y > u x$ ) (is -  $\longleftrightarrow$  ?alt)*

**proof**  
**assume** *lns: local-nonsatiation carrier relation*  
**have**  $\forall a b. a \succ b \longrightarrow u a > u b$   
**by** (*metis less-eq-real-def util-def x-y-in-carrier*)  
**then show** ?alt  
**by** (*meson lns local-nonsatiation-def*)

**next**  
**assume** *lns: ?alt*  
**show** *local-nonsatiation carrier relation*  
**proof**(*rule lns-normI*)  
**fix** *x* **and** *e::real*  
**assume** *x-in: x  $\in$  carrier*  
**assume** *e: e > 0*  
**have**  $\forall x \in \text{carrier}. \forall e > 0. \exists y \in \text{carrier}. \text{norm } (y - x) \leq e \wedge y \succ x$   
**by** (*meson less-eq-real-def linorder-not-less lns util-def*)  
**have**  $\exists y \in \text{carrier}. \text{norm } (y - x) \leq e \wedge u y > u x$   
**using** *e x-in lns by blast*  
**then show**  $\exists y \in \text{carrier}. \text{norm } (y - x) \leq e \wedge y \succ x$   
**by** (*meson compl not-less util-def x-in*)

```

qed
qed

end

lemma finite-carrier-rpr-iff-u:
  assumes finite carrier
    and (relation::'a relation)  $\subseteq$  carrier  $\times$  carrier
  shows rational-preference carrier relation  $\longleftrightarrow$  ( $\exists$  u. ordinal-utility carrier relation
u)
proof
  assume rational-preference carrier relation
  then show  $\exists$  u. ordinal-utility carrier relation u
    by (simp add: asms(1) fnt-carrier-exists-util-fun)
next
  assume  $\exists$  u. ordinal-utility carrier relation u
  then show rational-preference carrier relation
    by (metis (full-types) order-on-defs(1) ordinal-utility.util-imp-refl
ordinal-utility.util-imp-total ordinal-utility.util-imp-trans ordinal-utility-def
preference.intro rational-preference-axioms-def rational-preference-def)
qed

end

```

## 5 Consumers

Consumption sets

```

theory Consumers
  imports
    HOL-Analysis.Multivariate-Analysis
    ../Syntax
begin

```

### 5.1 Pre Arrow-Debreu consumption set

It turns out that the First Welfare Theorem does not require any particular limitations on the consumption set

```

locale pre-arrow-debreu-consumption-set =
  fixes consumption-set :: ('a::euclidean-space) set
  assumes  $x \in$  (UNIV:: 'a set)  $\implies$   $x \in$  consumption-set
begin
end

```



## 5.2 Arrow-Debreu model consumption set

The Arrow-Debreu model consumption set includes more and stricter assumptions which are necessary for further results.

```

locale gen-pre-arrow-debreu-consum-set =
  fixes consumption-set :: ('a::ordered-euclidean-space) set
begin

end

locale arrow-debreu-consum-set =
  fixes consumption-set :: ('a::ordered-euclidean-space) set
  assumes r-plus: consumption-set  $\subseteq$   $\{(x::'a). x \geq 0\}$ 
  assumes closed: closed consumption-set
  assumes convex: convex consumption-set
  assumes non-empty: consumption-set  $\neq$   $\{\}$ 
  assumes  $\forall M \in$  consumption-set.  $(\forall x > M. x \in$  consumption-set)
begin

lemma x-larger-0:  $x \in$  consumption-set  $\implies x \geq 0$ 
  using r-plus by auto

lemma larger-in-consump-set:
   $x \in$  consumption-set  $\wedge y \geq x \implies y \in$  consumption-set
  using arrow-debreu-consum-set-axioms arrow-debreu-consum-set-def
  dual-order.order-iff-strict by fastforce

end

end

```

```

theory Common
  imports
    ../Preferences
    ../Utility-Functions
    ../Argmax
begin

```

## 6 Pareto Ordering

Allows us to define a Pareto Ordering.

```

locale pareto-ordering =
  fixes agents :: 'i set
  fixes U :: 'i  $\Rightarrow$  'a  $\Rightarrow$  real
begin

```

**notation**  $U (U[-])$

**definition** *pareto-dominating* (**infix**  $\succ$ *Pareto* 60)

**where**

$$\begin{aligned} X \succ \text{Pareto } Y &\iff \\ &(\forall i \in \text{agents}. U[i] (X i) \geq U[i] (Y i)) \wedge \\ &(\exists i \in \text{agents}. U[i] (X i) > U[i] (Y i)) \end{aligned}$$

**lemma** *trans-strict-pareto*:  $X \succ \text{Pareto } Y \implies Y \succ \text{Pareto } Z \implies X \succ \text{Pareto } Z$

**proof** –

**assume**  $a1$ :  $X \succ \text{Pareto } Y$

**assume**  $Y \succ \text{Pareto } Z$

**then have**  $f\beta$ :  $\forall i \in \text{agents}. U[i] (Z i) \leq U[i] (X i)$

**by** (*meson a1 order-trans pareto-dominating-def*)

**moreover have**  $\exists i \in \text{agents}. \neg U[i] (X i) \leq U[i] (Y i)$

**using**  $a1$  *pareto-dominating-def* **by** *fastforce*

**ultimately show** *?thesis*

**by** (*metis*  $\langle Y \succ \text{Pareto } Z \rangle$  *less-eq-real-def pareto-dominating-def*)

**qed**

**lemma** *anti-sym-strict-pareto*:  $X \succ \text{Pareto } Y \implies \neg Y \succ \text{Pareto } X$

**using** *pareto-dominating-def* **by** *auto*

**end**

## 6.1 Budget constraint

Definition returns all affordable bundles given wealth  $W$

$f$  is a function that computes the value given a bundle

**definition** *budget-constraint*

**where**

$$\text{budget-constraint } f \ S \ W = \{x \in S. f \ x \leq W\}$$

## 6.2 Feasibility

**definition** *feasible-private-ownership*

**where**

$$\begin{aligned} \text{feasible-private-ownership } A \ F \ \mathcal{E} \ C_s \ P_s \ X \ Y &\iff \\ &(\sum_{i \in A}. X \ i) \leq (\sum_{i \in A}. \mathcal{E} \ i) + (\sum_{j \in F}. Y \ j) \wedge \\ &(\forall i \in A. X \ i \in C_s) \wedge (\forall j \in F. Y \ j \in P_s \ j) \end{aligned}$$

**lemma** *feasible-private-ownershipD*:

**assumes** *feasible-private-ownership*  $A \ F \ \mathcal{E} \ C_s \ P_s \ X \ Y$

**shows**  $(\sum_{i \in A}. X \ i) \leq (\sum_{i \in A}. \mathcal{E} \ i) + (\sum_{j \in F}. Y \ j)$

**and**  $(\forall i \in A. X \ i \in C_s)$  **and**  $(\forall j \in F. Y \ j \in P_s \ j)$

**using** *assms feasible-private-ownership-def* **apply** *blast*

**by** (*meson assms feasible-private-ownership-def*)

(*meson assms feasible-private-ownership-def*)

end

**theory** *Exchange-Economy*

**imports**

../Preferences

../Utility-Functions

../Argmax

Consumers

Common

**begin**

## 7 Exchange Economy

Define the exchange economy model

**locale** *exchange-economy* =

**fixes** *consumption-set* :: ('a::ordered-euclidean-space) set

**fixes** *agents* :: 'i set

**fixes**  $\mathcal{E}$  :: 'i  $\Rightarrow$  'a

**fixes** *Pref* :: 'i  $\Rightarrow$  'a relation

**fixes** *U* :: 'i  $\Rightarrow$  'a  $\Rightarrow$  real

**assumes** *cons-set-props*: pre-arrow-debreu-consumption-set *consumption-set*

**assumes** *agent-props*:  $i \in \text{agents} \implies \text{eucl-ordinal-utility } \text{consumption-set } (\text{Pref } i) (U i)$

**assumes** *finite-agents*: *finite agents* **and**  $\text{agents} \neq \{\}$

**sublocale** *exchange-economy*  $\subseteq$  *pareto-ordering agents U*

.

**context** *exchange-economy*

**begin**

**context**

**begin**

**notation** *U* (*U*[-])

**notation** *Pref* (*Pr*[-])

**notation**  $\mathcal{E}$  ( $\mathcal{E}$ [-])

**lemma** *base-pref-is-ord-eucl-rpr*:  $i \in \text{agents} \implies \text{rational-preference } \text{consumption-set } \text{Pr}[i]$

**by** (*meson exchange-economy.agent-props exchange-economy-axioms ord-eucl-utility-imp-rpr real-vector-rpr.have-rpr*)

**private abbreviation** *calculate-value*

**where**

*calculate-value*  $P x \equiv P \cdot x$

## 7.1 Feasibility

**definition** *feasible-allocation*

**where**

*feasible-allocation*  $A E \longleftrightarrow$   
 $(\sum_{i \in \text{agents.}} A i) \leq (\sum_{i \in \text{agents.}} E i)$

## 7.2 Pareto optimality

**definition** *pareto-optimal-endow*

**where**

*pareto-optimal-endow*  $X E \longleftrightarrow$   
 $(\text{feasible-allocation } X E \wedge$   
 $(\nexists X'. \text{feasible-allocation } X' E \wedge X' \succ \text{Pareto } X))$

## 7.3 Competitive Equilibrium in Exchange Economy

Competitive Equilibrium or Walrasian Equilibrium definition.

**definition** *comp-equilib-endow*

**where**

*comp-equilib-endow*  $P X E \equiv$   
*feasible-allocation*  $X E \wedge$   
 $(\forall i \in \text{agents. } X i \in \text{arg-max-set } U[i]$   
 $(\text{budget-constraint } (\text{calculate-value } P) \text{ consumption-set } (P \cdot E i)))$

## 7.4 Lemmas for final result

**lemma** *utility-function-def[iff]*:

**assumes**  $i \in \text{agents}$

**shows**  $U[i] x \geq U[i] y \longleftrightarrow x \succeq [\text{Pr}[i]] y$

**proof**

**have** *ordinal-utility consumption-set*  $(\text{Pref } i) (U[i])$

**using** *agent-props assms eucl-ordinal-utility-def* **by** *auto*

**then show**  $U[i] y \leq U[i] x \implies x \succeq [\text{Pref } i] y$

**by** (*meson UNIV-I cons-set-props ordinal-utility.util-def-conf*  
*pre-arrow-debreu-consumption-set-def*)

**next**

**show**  $x \succeq [\text{Pref } i] y \implies U[i] y \leq U[i] x$

**by** (*meson agent-props assms ordinal-utility-def eucl-ordinal-utility-def*)

**qed**

**lemma** *budget-constraint-is-feasible*:

**assumes**  $i \in \text{agents}$

**assumes**  $X \in (\text{budget-constraint } (\text{calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i]))$

**shows**  $P \cdot X \leq P \cdot \mathcal{E}[i]$

**using** *budget-constraint-def assms*

by (*simp add: budget-constraint-def*)

**lemma** *arg-max-set-therefore-no-better* :

**assumes**  $i \in \text{agents}$

**assumes**  $x \in \text{arg-max-set } U[i] \text{ (budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i]))$

**shows**  $U[i] \ y > U[i] \ x \longrightarrow y \notin \text{budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i])$

**by** (*meson no-better-in-s assms*)

Since we need no restriction on the consumption set for the First Welfare Theorem

**lemma** *consumption-set-member*:  $\forall x. x \in \text{consumption-set}$

**proof** –

**have**  $\bigwedge(x::'a). x \in \text{consumption-set}$

**using** *cons-set-props pre-arrow-debreu-consumption-set-def*

**by** (*simp add: pre-arrow-debreu-consumption-set-def*)

**thus** *?thesis*

**by** *blast*

**qed**

Under the assumption of Local non-satiation, agents will utilise their entire budget.

**lemma** *argmax-entire-budget* :

**assumes**  $i \in \text{agents}$

**assumes** *local-nonsatiation consumption-set Pr*[ $i$ ]

**assumes**  $X \in \text{arg-max-set } U[i] \text{ (budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i]))$

**shows**  $P \cdot X = P \cdot \mathcal{E}[i]$

**proof** –

**have** *leq* :  $(P \cdot X) \leq (P \cdot \mathcal{E}[i])$

**proof** –

**have**  $X \in \text{budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i])$

**using** *argmax-sol-in-s*[of  $X \ U[i]$ ] *budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i])]*

*assms* **by** *auto*

**thus** *?thesis*

**using** *assms(1) budget-constraint-is-feasible* **by** *blast*

**qed**

**have** *not-less*:  $\neg(P \cdot X < P \cdot \mathcal{E}[i])$

**proof**

**assume** *cpos*:  $(P \cdot X) < (P \cdot \mathcal{E}[i])$

**define** *lesS* **where**  $\text{lesS} = \{x. P \cdot x < P \cdot \mathcal{E}[i]\}$

**obtain** *e* **where**

$e: 0 < e \text{ ball } X \ e \subseteq \text{lesS}$

**by** (*metis cpos lesS-def mem-Collect-eq open-contains-ball-eq open-halfspace-lt*)

**obtain** *Y* **where**

$Y: Y \succ_{[\text{Pref } i]} X \ Y \in \text{ball } X \ e$

```

    using e consumption-set-member assms by blast
  have  $Y \in \text{consumption-set}$ 
    using consumption-set-member by blast
  hence  $Y \in \text{budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i])$ 
    using budget-constraint-def e lesS-def
      less-eq-real-def  $Y$  by fastforce
  thus False
    by (meson assms  $Y$  all-leq utility-function-def)
qed
show ?thesis
  using leq not-less by auto
qed

```

All bundles that would be strictly preferred to any argmax result, are more expensive.

**lemma** *pref-more-expensive:*

```

  assumes  $i \in \text{agents}$ 
    assumes  $x \in \text{arg-max-set } U[i] \text{ (budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i]))$ 
    assumes  $U[i] \ y > U[i] \ x$ 
    shows  $y \cdot P > P \cdot \mathcal{E}[i]$ 
proof (rule ccontr)
  assume cpos :  $\neg(y \cdot P > P \cdot \mathcal{E}[i])$ 
  then have xp-leq :  $y \cdot P \leq P \cdot \mathcal{E}[i]$ 
    by auto
  hence  $x \in \text{budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i])$ 
    using argmax-sol-in-s[of  $x \ U[i] \ \text{budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i])$ ]
      assms by auto
  hence xp-in:  $y \in \text{budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i])$ 
  proof -
    have  $P \cdot y \leq P \cdot \mathcal{E}[i]$ 
      by (metis xp-leq inner-commute)
    then show ?thesis
      using consumption-set-member by (simp add: budget-constraint-def)
  qed
  hence  $y \succ[\text{Pref } i] \ x$ 
    using arg-max-set-therefore-no-better assms by blast
  hence  $y \succ[\text{Pref } i] \ x \wedge y \in \text{budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i])$ 
    using xp-in by blast
  hence  $x \notin \text{arg-max-set } U[i] \text{ (budget-constraint (calculate-value } P) \text{ consumption-set } (P \cdot \mathcal{E}[i]))$ 
    by (meson assms exchange-economy.arg-max-set-therefore-no-better
      exchange-economy-axioms)
  then show False
    using assms(2) by auto
qed

```

Greater or equal utility implies greater or equal price.

**lemma** *same-util-is-equal-or-more-expensive*:

**assumes**  $i \in \text{agents}$

**assumes** *local-nonsatiation consumption-set*  $Pr[i]$

**assumes**  $x \in \text{arg-max-set } U[i]$  (*budget-constraint (calculate-value P) consumption-set*  $(P \cdot \mathcal{E}[i])$ )

**assumes**  $U[i] y \geq U[i] x$

**shows**  $y \cdot P \geq P \cdot \mathcal{E}[i]$

**proof** –

**have** *not-in*:  $y \notin \text{arg-max-set } U[i]$  (*budget-constraint (calculate-value P) consumption-set*  $(P \cdot \mathcal{E}[i])$ )

$\implies y \cdot P > P \cdot \mathcal{E}[i]$

**proof** –

**assume**  $y \notin \text{arg-max-set } U[i]$  (*budget-constraint (calculate-value P) consumption-set*  $(P \cdot \mathcal{E}[i])$ )

**then have**  $y \notin \text{budget-constraint (calculate-value P) consumption-set } (P \cdot \mathcal{E}[i])$

**by** (*meson assms leq-all-in-sol assms*)

**then show** *?thesis*

**by** (*simp add: budget-constraint-def inner-commute consumption-set-member*)

**qed**

**show** *?thesis*

**by** (*metis argmax-entire-budget not-in assms(1,2,3)*

*dual-order.order-iff-strict inner-commute*)

**qed**

**lemma** *all-in-argmax-same-price*:

**assumes**  $i \in \text{agents}$

**assumes** *local-nonsatiation consumption-set*  $Pr[i]$

**assumes**  $x \in \text{arg-max-set } U[i]$  (*budget-constraint (calculate-value P) consumption-set*  $(P \cdot \mathcal{E}[i])$ )

**and**  $y \in \text{arg-max-set } U[i]$  (*budget-constraint (calculate-value P) consumption-set*  $(P \cdot \mathcal{E}[i])$ )

**shows**  $P \cdot x = P \cdot y$

**using** *argmax-entire-budget assms(1) assms(2) assms(3) assms(4)* **by** *presburger*

All rationally acting agents (which is every agent by assumption) will not decrease his utility

**lemma** *individual-rationalism* :

**assumes** *comp-equilib-endow*  $P X \mathcal{E}$

**shows**  $\forall i \in \text{agents}. X i \succeq_{[Pref i]} \mathcal{E}[i]$

**by** (*metis pref-more-expensive comp-equilib-endow-def assms inner-commute less-irrefl not-le utility-function-def*)

**lemma** *walras-law-per-agent* :

**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{local-nonsatiation consumption-set } Pr[i]$

**assumes** *comp-equilib-endow*  $P X \mathcal{E}$

**shows**  $\forall i \in \text{agents}. P \cdot X i = P \cdot \mathcal{E}[i]$

**by** (*meson argmax-entire-budget comp-equilib-endow-def assms*)

Walras Law holds in our Exchange Economy model. It states that in an equilibrium, demand equals supply

**lemma** *walras-law*:

**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{local-nonsatiation consumption-set } Pr[i]$   
**assumes** *comp-equilib-endow*  $P \ X \ \mathcal{E}$   
**shows**  $(\sum_{i \in \text{agents}} P \cdot (X \ i)) - (\sum_{i \in \text{agents}} P \cdot \mathcal{E}[i]) = 0$   
**using** *assms walras-law-per-agent* **by** *auto*

**lemma** *inner-with-ge-0*:  $(P::(\text{real}, 'n::\text{finite}) \text{ vec}) > 0 \implies A \geq B \implies P \cdot A \geq P \cdot B$

**by** (*metis dual-order.order-iff-strict inner-commute interval-inner-leI(2) ord-class.atLeastAtMost-iff*)

## 7.5 First Welfare Theorem in Exchange Economy

We prove the first welfare theorem in our Exchange Economy model.

**theorem** *first-welfare-theorem-exchange*:

**assumes** *lms* :  $\bigwedge i. i \in \text{agents} \implies \text{local-nonsatiation consumption-set } Pr[i]$   
**and** *price-cond*:  $Price > 0$   
**assumes** *equilibrium* : *comp-equilib-endow*  $Price \ \mathcal{X} \ \mathcal{E}$   
**shows** *pareto-optimal-endow*  $\mathcal{X} \ \mathcal{E}$

**proof** (*rule ccontr*)

**assume** *neg-ass* :  $\neg \text{pareto-optimal-endow } \mathcal{X} \ \mathcal{E}$

**have** *equili-feasible* : *feasible-allocation*  $\mathcal{X} \ \mathcal{E}$

**using** *comp-equilib-endow-def equilibrium*

**by** (*simp add: comp-equilib-endow-def*)

**have** *price-g-zero* :  $Price > 0$

**by** (*simp add: price-cond*)

**obtain**  $Y$  **where**

*xprime-pareto*: *feasible-allocation*  $Y \ \mathcal{E} \wedge$

$(\forall i \in \text{agents}. U[i] (Y \ i) \geq U[i] (\mathcal{X} \ i)) \wedge$

$(\exists i \in \text{agents}. U[i] (Y \ i) > U[i] (\mathcal{X} \ i))$

**using** *equili-feasible neg-ass pareto-dominating-def*

*pareto-optimal-endow-def* **by** *auto*

**have** *is-feasible* : *feasible-allocation*  $Y \ \mathcal{E}$

**using** *xprime-pareto* **by** *blast*

**have** *all-great-eq-value* :  $\forall i \in \text{agents}. Price \cdot (Y \ i) \geq Price \cdot (\mathcal{X} \ i)$

**proof**

**fix**  $i$

**assume**  $i \in \text{agents}$

**show**  $Price \cdot (Y \ i) \geq Price \cdot (\mathcal{X} \ i)$

**proof** –

**have** *x-in-agmx* :  $(\mathcal{X} \ i) \in \text{arg-max-set } U[i] \text{ (budget-constraint (calculate-value } Price) \text{ consumption-set (Price} \cdot \mathcal{E}[i]))}$

**by** (*meson*  $\langle i \in \text{agents} \rangle$  *comp-equilib-endow-def equilibrium*)

**have**  $(U[i] (\mathcal{X} \ i)) - U[i] (Y \ i) \leq 0$

**using**  $\langle i \in \text{agents} \rangle$  *xprime-pareto* **by** *auto*

**hence**  $Price \cdot (\mathcal{X} \ i) - Price \cdot (Y \ i) \leq 0$



```

    by (metis ⟨i ∈ agents⟩ argmax-entire-budget diff-le-0-iff-le x-in-agmx
        inner-commute lns same-util-is-equal-or-more-expensive)
  then show ?thesis
    by auto
qed
qed
have ex-greater-value : ∃ i ∈ agents. Price · (Y i) > Price · (X i)
proof (rule ccontr)
  assume a1 : ¬(∃ i ∈ agents. Price · (Y i) > Price · (X i))
  obtain i where
    obt-witness : i ∈ agents U[i] (Y i) > ( U[i]) (X i)
  using xprime-pareto by blast
  have Price · Y i ≠ Price · X i
  proof -
    have Price · Y i > Price · E i
      by (metis pref-more-expensive comp-equilib-endow-def
          equilibrium inner-commute obt-witness(1) obt-witness(2))
    have Price · E i = Price · X i
      using equilibrium lns obt-witness(1) walras-law-per-agent by auto
    then show ?thesis
      using ⟨Price · E i < Price · Y i⟩ by linarith
  qed
  then show False
    using a1 all-great-eq-value obt-witness(1) by fastforce
qed
have dominating-more-exp : Price · (∑ i∈agents. Y i) > Price · (∑ i∈agents.
X i)
proof -
  have mp-rule : (∑ i∈agents. Price · Y i) > (∑ i∈agents. Price · X i) ⇒
?thesis
    by (simp add: inner-sum-right)
  have (∑ i∈agents. Price · Y i) > (∑ i∈agents. Price · X i)
  by (simp add: all-great-eq-value finite-agents ex-greater-value sum-strict-mono-ex1)
  thus Price · (∑ i∈agents. Y i) > Price · (∑ i∈agents. X i)
    using mp-rule by blast
qed
have equili-walras-law : Price · (∑ i∈agents. X i) = Price · (∑ i∈agents. E[i])
  by (metis (mono-tags) eq-iff-diff-eq-0 equilibrium
      inner-sum-right lns walras-law)
have dominating-feasible : Price · (∑ i∈agents. X i) ≥ Price · (∑ i∈agents. Y
i)
  by (metis atLeastAtMost-iff dual-order.order-iff-strict equili-walras-law
      feasible-allocation-def inner-commute interval-inner-leI(1) is-feasible price-g-zero)
show False
  using dominating-more-exp equili-walras-law dominating-feasible
  by linarith
qed

```

Monotone preferences can be used instead of local non-satiation. Many

textbooks etc. do not introduce the concept of local non-satiation and use monotonicity instead.

**corollary** *first-welfare-exch-thm-monot*:

**assumes**  $\forall M \in \text{carrier}. (\forall x > M. x \in \text{carrier})$

**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{monotone-preference consumption-set } Pr[i]$

**and price-cond**:  $Price > 0$

**assumes** *comp-equilib-endow*  $Price \mathcal{X} \mathcal{E}$

**shows** *pareto-optimal-endow*  $\mathcal{X} \mathcal{E}$

**by** (*meson assms exchange-economy.consumption-set-member*

*first-welfare-theorem-exchange exchange-economy-axioms unbounded-above-mono-imp-lns*)

**end**

**end**

**end**

## 8 Pre Arrow-Debreu model

Model similar to Arrow-Debreu model but with fewer assumptions, since we only need assumptions strong enough to proof the First Welfare Theorem.

**theory** *Private-Ownership-Economy*

**imports**

*../Preferences*

*../Preferences*

*../Utility-Functions*

*../Argmax*

*Consumers*

*Common*

**begin**

**locale** *pre-arrow-debreu-model* =

**fixes** *production-sets* ::  $'f \Rightarrow ('a :: \text{ordered-euclidean-space}) \text{ set}$

**fixes** *consumption-set* ::  $'a \text{ set}$

**fixes** *agents* ::  $'i \text{ set}$

**fixes** *firms* ::  $'f \text{ set}$

**fixes**  $\mathcal{E} :: 'i \Rightarrow 'a (\mathcal{E}[-])$

**fixes** *Pref* ::  $'i \Rightarrow 'a \text{ relation } (Pr[-])$

**fixes**  $U :: 'i \Rightarrow 'a \Rightarrow \text{real } (U[-])$

**fixes**  $\Theta :: 'i \Rightarrow 'f \Rightarrow \text{real } (\Theta[-,-])$

**assumes** *cons-set-props*: *pre-arrow-debreu-consumption-set consumption-set*

**assumes** *agent-props*:  $i \in \text{agents} \implies \text{eucl-ordinal-utility consumption-set } (Pr[i])$   
( $U[i]$ )

**assumes** *firms-comp-owned*:  $j \in \text{firms} \implies (\sum i \in \text{agents}. \Theta[i,j]) = 1$

**assumes** *finite-nonepty-agents*: *finite agents* **and**  $\text{agents} \neq \{\}$

**sublocale** *pre-arrow-debreu-model*  $\subseteq$  *pareto-ordering agents U*

**context** *pre-arrow-debreu-model*  
**begin**

No restrictions on consumption set needed

**lemma** *all-larger-zero-in-csset*:  $\forall x. x \in \text{consumption-set}$   
**using** *cons-set-props pre-arrow-debreu-consumption-set-def* **by** *blast*

**context**  
**begin**

Calculate wealth of individual i in context of Private Ownership economy.

**private abbreviation** *poe-wealth*

**where**

$$\text{poe-wealth } P \ i \ Y \equiv P \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms.}} \Theta[i,j] *_{\mathbb{R}} (P \cdot Y \ j))$$

## 8.1 Feasibility

**private abbreviation** *feasible*

**where**

*feasible X Y*  $\equiv$  *feasible-private-ownership agents firms E consumption-set production-sets X Y*

**private abbreviation** *calculate-value*

**where**

$$\text{calculate-value } P \ x \equiv P \cdot x$$

## 8.2 Profit maximisation

In a production economy we need to specify profit maximisation.

**definition** *profit-maximisation*

**where**

$$\text{profit-maximisation } P \ S = \text{arg-max-set } (\lambda x. P \cdot x) \ S$$

## 8.3 Competitive Equilibrium

Competitive equilibrium in context of production economy with private ownership. This includes the profit maximisation condition.

**definition** *competitive-equilibrium*

**where**

*competitive-equilibrium P X Y*  $\longleftrightarrow$  *feasible X Y*  $\wedge$   
 $(\forall j \in \text{firms. } (Y \ j) \in \text{profit-maximisation } P \ (\text{production-sets } j)) \wedge$   
 $(\forall i \in \text{agents. } (X \ i) \in \text{arg-max-set } U[i] \ (\text{budget-constraint } (\text{calculate-value } P) \text{ consumption-set } (\text{poe-wealth } P \ i \ Y)))$

**lemma** *competitive-equilibriumD* [*dest*]:  
**assumes** *competitive-equilibrium P X Y*  
**shows** *feasible X Y*  $\wedge$   
 $(\forall j \in \text{firms}. (Y j) \in \text{profit-maximisation } P (\text{production-sets } j)) \wedge$   
 $(\forall i \in \text{agents}. (X i) \in \text{arg-max-set } U[i] (\text{budget-constraint } (\text{calculate-value } P) \text{ consumption-set } (\text{poe-wealth } P i Y)))$   
**using** *assms* **by** (*simp add: competitive-equilibrium-def*)

**lemma** *compet-max-profit*:  
**assumes**  $j \in \text{firms}$   
**assumes** *competitive-equilibrium P X Y*  
**shows**  $Y j \in \text{profit-maximisation } P (\text{production-sets } j)$   
**using** *assms(1) assms(2)* **by** *blast*

## 8.4 Pareto Optimality

**definition** *pareto-optimal*  
**where**  
 $\text{pareto-optimal } X Y \iff$   
 $(\text{feasible } X Y \wedge$   
 $(\nexists X' Y'. \text{feasible } X' Y' \wedge X' \succ \text{Pareto } X))$

**lemma** *pareto-optimalI[intro]*:  
**assumes** *feasible X Y*  
**and**  $\nexists X' Y'. \text{feasible } X' Y' \wedge X' \succ \text{Pareto } X$   
**shows** *pareto-optimal X Y*  
**using** *pareto-optimal-def assms(1) assms(2)* **by** *blast*

**lemma** *pareto-optimalD[dest]*:  
**assumes** *pareto-optimal X Y*  
**shows** *feasible X Y* **and**  $\nexists X' Y'. \text{feasible } X' Y' \wedge X' \succ \text{Pareto } X$   
**using** *pareto-optimal-def assms* **by** *auto*

**lemma** *util-fun-def-holds*:  $i \in \text{agents} \implies x \succeq_{[Pr[i]]} y \iff U[i] x \geq U[i] y$   
**by** (*meson agent-props all-larger-zero-in-csset eucl-ordinal-utility-def ordinal-utility-def*)

**lemma** *base-pref-is-ord-eucl-rpr*:  $i \in \text{agents} \implies \text{rational-preference consumption-set } Pr[i]$   
**using** *agent-props ord-eucl-utility-imp-rpr real-vector-rpr.have-rpr* **by** *blast*

**lemma** *prof-max-ge-all-in-pset*:  
**assumes**  $j \in \text{firms}$   
**assumes**  $Y j \in \text{profit-maximisation } P (\text{production-sets } j)$   
**shows**  $\forall y \in \text{production-sets } j. P \cdot Y j \geq P \cdot y$   
**using** *all-leq assms(2) profit-maximisation-def* **by** *fastforce*

## 8.5 Lemmas for final result

Strictly preferred bundles are strictly more expensive.

**lemma** *all-preferred-are-more-expensive*:

**assumes** *i-agt*:  $i \in \text{agents}$

**assumes** *equil*: *competitive-equilibrium*  $P \ \mathcal{X} \ \mathcal{Y}$

**assumes**  $z \in \text{consumption-set}$

**assumes**  $(U \ i) \ z > (U \ i) \ (\mathcal{X} \ i)$

**shows**  $z \cdot P > P \cdot (\mathcal{X} \ i)$

**proof** (*rule ccontr*)

**assume** *neg-as* :  $\neg(z \cdot P > P \cdot (\mathcal{X} \ i))$

**have** *xp-leq* :  $z \cdot P \leq P \cdot (\mathcal{X} \ i)$

**using**  $\langle \neg z \cdot P > P \cdot (\mathcal{X} \ i) \rangle$  **by** *auto*

**have** *x-in-argmax*:  $(\mathcal{X} \ i) \in \text{arg-max-set } U[i]$  (*budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y)*)

**using** *equil i-agt* **by** *blast*

**hence** *x-in*:  $\mathcal{X} \ i \in (\text{budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y)})$

**using** *argmax-sol-in-s [of (X i) U[i] budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y)]*

**by** *blast*

**hence** *z-in-budget*:  $z \in (\text{budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y)})$

**proof** –

**have** *z-leq-endow*:  $P \cdot z \leq P \cdot (\mathcal{X} \ i)$

**by** (*metis xp-leq inner-commute*)

**have** *z-in-cons*:  $z \in \text{consumption-set}$

**using** *assms* **by** *auto*

**then show** *?thesis*

**using** *x-in budget-constraint-def z-leq-endow*

**proof** –

**have**  $\forall r. P \cdot \mathcal{X} \ i \leq r \longrightarrow P \cdot z \leq r$

**using** *z-leq-endow* **by** *linarith*

**then show** *?thesis*

**using** *budget-constraint-def x-in z-in-cons*

**by** (*simp add: budget-constraint-def*)

**qed**

**qed**

**have** *nex-prop*:  $\nexists e. e \in (\text{budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y)}) \wedge$

$U[i] \ e > U[i] \ (\mathcal{X} \ i)$

**using** *no-better-in-s[of X i U[i] budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y)]*

*x-in-argmax* **by** *blast*

**have**  $z \in \text{budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y)} \wedge U[i] \ z > U[i] \ (\mathcal{X} \ i)$

**using** *assms z-in-budget* **by** *blast*

**thus** *False* **using** *nex-prop*

**by** *blast*

**qed**

Given local non-satiation, argmax will use the entire budget.

**lemma** *am-utilises-entire-bgt*:

**assumes** *i-agts*:  $i \in \text{agents}$

**assumes** *lns* : *local-nonsatiation consumption-set*  $Pr[i]$

**assumes** *argmax-sol* :  $X \in \text{arg-max-set } U[i]$  (*budget-constraint* (*calculate-value*  $P$ ) *consumption-set* (*poe-wealth*  $P$   $i$   $Y$ ))

**shows**  $P \cdot X = P \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms.}} \Theta[i,j] *_R (P \cdot Y j))$

**proof** –

**let**  $?wlt = P \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms.}} \Theta[i,j] *_R (P \cdot Y j))$

**let**  $?bc = \text{budget-constraint} (\text{calculate-value } P) \text{ consumption-set} (\text{poe-wealth } P \ i \ Y)$

**have**  $X \in \text{budget-constraint} (\text{calculate-value } P) \text{ consumption-set} (\text{poe-wealth } P \ i \ Y)$

**using** *argmax-sol-in-s* [*of*  $X$   $U[i]$   $?bc$ ] *argmax-sol* **by** *blast*

**hence** *is-leq*:  $X \cdot P \leq (\text{poe-wealth } P \ i \ Y)$

**by** (*metis* (*mono-tags*, *lifting*) *budget-constraint-def* *inner-commute mem-Collect-eq*)

**have** *not-less*:  $\neg X \cdot P < (\text{poe-wealth } P \ i \ Y)$

**proof**

**assume** *neg*:  $X \cdot P < (\text{poe-wealth } P \ i \ Y)$

**have** *bgt-leq*:  $\forall x \in ?bc. U[i] X \geq U[i] x$

**using** *leq-all-in-sol* [*of*  $X$   $U[i]$   $?bc$ ]

*all-leq* [*of*  $X$   $U[i]$   $?bc$ ]

*argmax-sol* **by** *blast*

**define** *s-low* **where**

$s\text{-low} = \{x . P \cdot x < ?wlt\}$

**have**  $\exists e > 0. \text{ball } X \ e \subseteq s\text{-low}$

**proof** –

**have** *x-in-budget*:  $P \cdot X < ?wlt$

**by** (*metis* *inner-commute neg*)

**have** *s-low-open*: *open*  $s\text{-low}$

**using** *open-halfspace-lt s-low-def* **by** *blast*

**then show** *?thesis*

**using** *s-low-open open-contains-ball-eq* *s-low-def x-in-budget* **by** *blast*

**qed**

**obtain**  $e$  **where**

$e > 0$  **and**  $e$ :  $\text{ball } X \ e \subseteq s\text{-low}$

**using**  $\langle \exists e > 0. \text{ball } X \ e \subseteq s\text{-low} \rangle$  **by** *blast*

**obtain**  $y$  **where**

*y-props*:  $y \in \text{ball } X \ e \ \succ_{[Pref \ i]} X$

**using**  $\langle 0 < e \rangle$  *all-larger-zero-in-csset lns* **by** *blast*

**have**  $y \in \text{budget-constraint} (\text{calculate-value } P) \text{ consumption-set} (\text{poe-wealth } P \ i \ Y)$

**proof** –

**have**  $y \in s\text{-low}$

**using**  $\langle y \in \text{ball } X \ e \rangle \ e$  **by** *blast*

**then show** *?thesis*

**by** (*simp add*: *s-low-def all-larger-zero-in-csset* *budget-constraint-def*)

```

qed
then show False
  using bgt-leq i-agts y-props(2) util-fun-def-holds by blast
qed
then show ?thesis
  by (metis inner-commute is-leq
    less-eq-real-def)
qed

```

**corollary** *x-equil-x-ext-budget:*

```

assumes i-agt: i ∈ agents
assumes lns : local-nonsatiation consumption-set Pr[i]
assumes equilibrium : competitive-equilibrium P X Y
shows P · X i = P · E[i] + (∑ j∈firms. Θ[i,j] *R (P · Y j))

```

**proof** –

```

have X i ∈ arg-max-set U[i] (budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y))
  using equilibrium i-agt by blast
then show ?thesis
  using am-utilises-entire-bgt i-agt lns by blast
qed

```

**lemma** *same-price-in-argmax :*

```

assumes i-agt: i ∈ agents
assumes lns : local-nonsatiation consumption-set Pr[i]
assumes x ∈ arg-max-set (U[i]) (budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y))
assumes y ∈ arg-max-set (U[i]) (budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y))
shows (P · x) = (P · y)
  using am-utilises-entire-bgt assms lns
  by (metis (no-types) am-utilises-entire-bgt assms(3) assms(4) i-agt lns)

```

Greater or equal utility implies greater or equal value.

**lemma** *utility-ge-price-ge :*

```

assumes agts: i ∈ agents
assumes lns : local-nonsatiation consumption-set Pr[i]
assumes equil: competitive-equilibrium P X Y
assumes geq: U[i] z ≥ U[i] (X i)
  and z ∈ consumption-set
shows P · z ≥ P · (X i)

```

**proof** –

```

let ?bc = (budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y))
have not-in : z ∉ arg-max-set (U[i]) ?bc ⇒
  P · z > (P · E[i]) + (∑ j∈(firms). (Θ[i,j] *R (P · Y j)))
proof –
  assume z ∉ arg-max-set (U[i]) ?bc
  moreover have X i ∈ arg-max-set (U[i]) ?bc

```

```

    using competitive-equilibriumD assms pareto-optimal-def
    by auto
    ultimately have  $z \notin \text{budget-constraint (calculate-value } P) \text{ consumption-set}$ 
    (poe-wealth  $P \ i \ Y$ )
    by (meson geq leq-all-in-sol)
    then show ?thesis
    using budget-constraint-def assms
    by (simp add: budget-constraint-def)
qed
have  $x\text{-in-argmax: } (X \ i) \in \text{arg-max-set } U[i] \ ?bc$ 
    using agts equil by blast
hence  $x\text{-in-budget: } (X \ i) \in \ ?bc$ 
    using argmax-sol-in-s [of  $(X \ i) \ U[i] \ ?bc$ ] by blast
have  $U[i] \ z = U[i] \ (X \ i) \implies P \cdot z \geq P \cdot (X \ i)$ 
proof(rule contrapos-pp)
  assume con-neg:  $\neg P \cdot z \geq P \cdot (X \ i)$ 
  then have  $P \cdot z < P \cdot (X \ i)$ 
    by linarith
  then have  $z\text{-in-argmax: } z \in \text{arg-max-set } U[i] \ ?bc$ 
proof -
  have  $P \cdot (X \ i) = P \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms.}} \Theta[i,j] *_R (P \cdot Y \ j))$ 
    using agts am-utilises-entire-bgt lns  $x\text{-in-argmax}$  by blast
  then show ?thesis
    by (metis (no-types) con-neg less-eq-real-def not-in)
qed
have  $z\text{-budget-utilisation: } P \cdot z = P \cdot (X \ i)$ 
    by (metis (no-types) agts am-utilises-entire-bgt lns  $x\text{-in-argmax}$   $z\text{-in-argmax}$ )
have  $P \cdot (X \ i) = P \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms.}} \Theta[i,j] *_R (P \cdot Y \ j))$ 
    using agts am-utilises-entire-bgt lns  $x\text{-in-argmax}$  by blast
show  $\neg U[i] \ z = U[i] \ (X \ i)$ 
    using  $z\text{-budget-utilisation}$  con-neg by linarith
qed
thus ?thesis
    by (metis (no-types) agts am-utilises-entire-bgt eq-iff eucl-less-le-not-le lns not-in
 $x\text{-in-argmax}$ )
qed

```

**lemma** *commutativity-sums-over-funs:*

```

fixes  $X :: 'x \text{ set}$ 
fixes  $Y :: 'y \text{ set}$ 
shows  $(\sum_{i \in X. \sum_{j \in Y.} (f \ i \ j *_R C \cdot g \ j)) = (\sum_{j \in Y.} \sum_{i \in X.} (f \ i \ j *_R C \cdot g \ j))$ 
    using Groups-Big.comm-monoid-add-class.sum.swap by auto

```

**lemma** *assoc-fun-over-sum:*

```

fixes  $X :: 'x \text{ set}$ 
fixes  $Y :: 'y \text{ set}$ 
shows  $(\sum_{j \in Y. \sum_{i \in X.} f \ i \ j *_R C \cdot g \ j) = (\sum_{j \in Y.} (\sum_{i \in X.} f \ i \ j) *_R C \cdot g \ j)$ 
    by (simp add: inner-sum-left scaleR-left.sum)

```

Walras' law in context of production economy with private ownership. That



is, in an equilibrium demand equals supply.

**lemma walras-law:**

**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{local-nonsatiation consumption-set } Pr[i]$   
**assumes**  $(\forall i \in \text{agents}. (X i) \in \text{arg-max-set } U[i] \text{ (budget-constraint (calculate-value } P) \text{ consumption-set (poe-wealth } P i Y)))$   
**shows**  $P \cdot (\sum i \in \text{agents}. (X i)) = P \cdot ((\sum i \in \text{agents}. \mathcal{E}[i]) + (\sum j \in \text{firms}. Y j))$   
**proof** –  
**have value-equal:**  $P \cdot (\sum i \in \text{agents}. (X i)) = P \cdot (\sum i \in \text{agents}. \mathcal{E}[i]) + (\sum i \in \text{agents}. \sum f \in \text{firms}. \Theta[i,f] *_R (P \cdot Y f))$   
**proof** –  
**have all-exhaust-bgt:**  $\forall i \in \text{agents}. P \cdot (X i) = P \cdot \mathcal{E}[i] + (\sum j \in \text{firms}. \Theta[i,j] *_R (P \cdot (Y j)))$   
**using** *assms am-utilises-entire-bgt by blast*  
**then show** *?thesis*  
**by** (*simp add:all-exhaust-bgt inner-sum-right sum.distrib*)  
**qed**  
**have eq-1:**  $(\sum i \in \text{agents}. \sum j \in \text{firms}. (\Theta[i,j] *_R P \cdot Y j)) = (\sum j \in \text{firms}. \sum i \in \text{agents}. (\Theta[i,j] *_R P \cdot Y j))$   
**using** *commutativity-sums-over-funs [of  $\Theta P Y$  firms agents] by blast*  
**hence eq-2:**  $P \cdot (\sum i \in \text{agents}. X i) = P \cdot (\sum i \in \text{agents}. \mathcal{E}[i]) + (\sum j \in \text{firms}. \sum i \in \text{agents}. \Theta[i,j] *_R P \cdot Y j)$   
**using** *value-equal by auto*  
**also have eq-3:**  $\dots = P \cdot (\sum i \in \text{agents}. \mathcal{E}[i]) + (\sum j \in \text{firms}. (\sum i \in \text{agents}. \Theta[i,j] *_R P \cdot Y j))$   
**using** *assoc-fun-over-sum[of  $\Theta P Y$  agents firms] by auto*  
**also have eq-4:**  $\dots = P \cdot (\sum i \in \text{agents}. \mathcal{E}[i]) + (\sum f \in \text{firms}. P \cdot Y f)$   
**using** *firms-comp-owned by auto*  
**have comp-wise-inner:**  $P \cdot (\sum i \in \text{agents}. X i) - (P \cdot (\sum i \in \text{agents}. \mathcal{E}[i]) - (\sum f \in \text{firms}. P \cdot Y f)) = 0$   
**using** *eq-1 eq-2 eq-3 eq-4 by linarith*  
**then show** *?thesis*  
**by** (*simp add: inner-right-distrib inner-sum-right*)  
**qed**

**lemma walras-law-in-compeq:**

**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{local-nonsatiation consumption-set } Pr[i]$   
**assumes** *competitive-equilibrium  $P X Y$*   
**shows**  $P \cdot ((\sum i \in \text{agents}. (X i)) - (\sum i \in \text{agents}. \mathcal{E}[i]) - (\sum j \in \text{firms}. Y j)) = 0$   
**proof** –  
**have**  $P \cdot (\sum i \in \text{agents}. (X i)) = P \cdot ((\sum i \in \text{agents}. \mathcal{E}[i]) + (\sum j \in \text{firms}. Y j))$   
**using** *assms(1) assms(2) walras-law by auto*  
**then show** *?thesis*  
**by** (*simp add: inner-diff-right inner-right-distrib*)  
**qed**

## 8.6 First Welfare Theorem

Proof of First Welfare Theorem in context of production economy with private ownership.

**theorem** *first-welfare-theorem-priv-own*:  
**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{local-nonsatiation consumption-set } Pr[i]$   
**and**  $Price > 0$   
**assumes** *competitive-equilibrium*  $Price \mathcal{X} \mathcal{Y}$   
**shows** *pareto-optimal*  $\mathcal{X} \mathcal{Y}$   
**proof** (*rule ccontr*)  
**assume** *neg-as*:  $\neg \text{pareto-optimal } \mathcal{X} \mathcal{Y}$   
**have** *equili-feasible* : *feasible*  $\mathcal{X} \mathcal{Y}$   
**using** *assms* **by** (*simp add: competitive-equilibrium-def*)  
**obtain**  $X' Y'$  **where**  
*xprime-pareto*: *feasible*  $X' Y' \wedge$   
 $(\forall i \in \text{agents}. U[i] (X' i) \geq U[i] (\mathcal{X} i)) \wedge$   
 $(\exists i \in \text{agents}. U[i] (X' i) > U[i] (\mathcal{X} i))$   
**using** *equili-feasible pareto-optimal-def*  
*pareto-dominating-def neg-as* **by** *auto*  
**have** *is-feasible*: *feasible*  $X' Y'$   
**using** *xprime-pareto* **by** *blast*  
**have** *xprime-leq-y*:  $\forall i \in \text{agents}. (Price \cdot (X' i) \geq$   
 $(Price \cdot \mathcal{E}[i]) + (\sum j \in (\text{firms}). \Theta[i,j] *_{\mathbb{R}} (Price \cdot \mathcal{Y} j)))$   
**proof**  
**fix**  $i$   
**assume** *as*:  $i \in \text{agents}$   
**have** *xprime-cons*:  $X' i \in \text{consumption-set}$   
**by** (*simp add: all-larger-zero-in-csset*)  
**have** *x-leq-xprime*:  $U[i] (X' i) \geq U[i] (\mathcal{X} i)$   
**using**  $\langle i \in \text{agents} \rangle$  *xprime-pareto* **by** *blast*  
**have** *lms-pref*: *local-nonsatiation consumption-set*  $Pr[i]$   
**using** *as* *assms* **by** *blast*  
**hence** *xprime-ge-x*:  $Price \cdot (X' i) \geq Price \cdot (\mathcal{X} i)$   
**using** *x-leq-xprime xprime-cons* *as* *assms* *utility-ge-price-ge* **by** *blast*  
**then show**  $Price \cdot (X' i) \geq (Price \cdot \mathcal{E}[i]) + (\sum j \in (\text{firms}). \Theta[i,j] *_{\mathbb{R}} (Price \cdot$   
 $\mathcal{Y} j))$   
**using** *xprime-ge-x*  $\langle i \in \text{agents} \rangle$  *lms-pref* *assms* *x-equil-x-ext-budget* **by** *fastforce*  
**qed**  
**have** *ex-greater-value* :  $\exists i \in \text{agents}. Price \cdot (X' i) > Price \cdot (\mathcal{X} i)$   
**proof**(*rule ccontr*)  
**assume** *cpos* :  $\neg(\exists i \in \text{agents}. Price \cdot (X' i) > Price \cdot (\mathcal{X} i))$   
**obtain**  $i$  **where**  
*obt-witness* :  $i \in \text{agents} (U[i] (X' i) > U[i] (\mathcal{X} i))$   
**using** *xprime-pareto* **by** *blast*  
**show** *False*  
**by** (*metis cpos all-larger-zero-in-csset all-preferred-are-more-expensive*  
*inner-commute obt-witness(1) obt-witness(2) assms(3)*)  
**qed**  
**have** *dom-g* :  $Price \cdot (\sum i \in \text{agents}. X' i) > Price \cdot (\sum i \in \text{agents}. (\mathcal{X} i))$  (*is - >*  
*- . ?x-sum*)  
**proof** -  
**have**  $(\sum i \in \text{agents}. Price \cdot X' i) > (\sum i \in \text{agents}. Price \cdot (\mathcal{X} i))$   
**by** (*metis (mono-tags, lifting) xprime-leq-y assms(1,3) ex-greater-value*)

*finite-nonepty-agents sum-strict-mono-ex1 x-equil-x-ext-budget*)

**thus**  $Price \cdot (\sum_{i \in agents} X' i) > Price \cdot ?x\text{-sum}$   
**by** (*simp add: inner-sum-right*)

**qed**

**let**  $?y\text{-sum} = (\sum_{j \in firms} \mathcal{Y} j)$   
**have** *equili-walras-law*:  $Price \cdot ?x\text{-sum} =$   
 $(\sum_{i \in agents} Price \cdot \mathcal{E}[i] + (\sum_{j \in firms} \Theta[i,j] *_R (Price \cdot \mathcal{Y} j)))$  (**is - = ?ws**)

**proof -**  
**have**  $\forall i \in agents. Price \cdot \mathcal{X} i = Price \cdot \mathcal{E}[i] + (\sum_{j \in firms} \Theta[i,j] *_R (Price \cdot \mathcal{Y} j))$   
**by** (*metis (no-types, lifting) assms(1,3) x-equil-x-ext-budget*)  
**then show** *?thesis*  
**by** (*simp add: inner-sum-right*)

**qed**

**also have** *remove-firm-pct*:  $\dots = Price \cdot (\sum_{i \in agents} \mathcal{E}[i]) + (Price \cdot ?y\text{-sum})$

**proof -**  
**have** *equals-inner-price:0*  $= Price \cdot (?x\text{-sum} - ((\sum_{i \in agents} \mathcal{E} i) + ?y\text{-sum}))$   
**by** (*metis (no-types) diff-diff-add assms(1,3) walras-law-in-compeq*)  
**have**  $Price \cdot ?x\text{-sum} = Price \cdot ((\sum_{i \in agents} \mathcal{E} i) + ?y\text{-sum})$   
**by** (*metis (no-types) equals-inner-price inner-diff-right right-minus-eq*)  
**then show** *?thesis*  
**by** (*simp add: equili-walras-law inner-right-distrib*)

**qed**

**have** *xp-l-yp*:  $(\sum_{i \in agents} X' i) \leq (\sum_{i \in agents} \mathcal{E}[i]) + (\sum_{f \in firms} Y' f)$   
**using** *is-feasible feasible-private-ownership-def* **by** *blast*

**hence** *yprime-sgr-y*:  $Price \cdot (\sum_{i \in agents} \mathcal{E}[i]) + Price \cdot (\sum_{f \in firms} Y' f) >$   
*?ws*

**proof -**  
**have**  $Price \cdot (\sum_{i \in agents} X' i) \leq Price \cdot ((\sum_{i \in agents} \mathcal{E}[i]) + (\sum_{j \in firms} Y' j))$   
**by** (*metis xp-l-yp atLeastAtMost-iff inner-commute interval-inner-leI(2) less-imp-le order-reft assms(2)*)  
**hence**  $?ws < Price \cdot ((\sum_{i \in agents} \mathcal{E} i) + (\sum_{j \in firms} Y' j))$   
**using** *dom-g equili-walras-law* **by** *linarith*  
**then show** *?thesis*  
**by** (*simp add: inner-right-distrib*)

**qed**

**have** *Y-is-optimum*:  $\forall j \in firms. \forall y \in production\text{-sets } j. Price \cdot \mathcal{Y} j \geq Price \cdot y$   
**using** *assms prof-max-ge-all-in-pset* **by** *blast*

**have** *yprime-in-prod-set*:  $\forall j \in firms. Y' j \in production\text{-sets } j$   
**using** *xprime-pareto* **by** (*simp add: feasible-private-ownership-def*)

**hence**  $\forall j \in firms. \forall y \in production\text{-sets } j. Price \cdot \mathcal{Y} j \geq Price \cdot y$   
**using** *Y-is-optimum* **by** *blast*

**hence** *Y-ge-yprime*:  $\forall j \in firms. Price \cdot \mathcal{Y} j \geq Price \cdot Y' j$   
**using** *yprime-in-prod-set* **by** *blast*

**hence** *yprime-p-leq-Y*:  $Price \cdot (\sum_{f \in firms} Y' f) \leq Price \cdot ?y\text{-sum}$   
**by** (*simp add: Y-ge-yprime inner-sum-right sum-mono*)

**then show** *False*  
**using** *remove-firm-pct yprime-sgr-y* **by** *linarith*

qed

Equilibrium cannot be Pareto dominated.

**lemma** *equilibria-dom-eachother:*

**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{local-nonsatiation consumption-set } Pr[i]$   
**and**  $Price > 0$

**assumes** *equil: competitive-equilibrium*  $Price \mathcal{X} \mathcal{Y}$

**shows**  $\nexists X' Y'. \text{competitive-equilibrium } P X' Y' \wedge X' \succ \text{Pareto } \mathcal{X}$

**proof** –

**have** *pareto-optimal*  $\mathcal{X} \mathcal{Y}$

**by** (*meson assms equil first-welfare-theorem-priv-own*)

**hence**  $\nexists X' Y'. \text{feasible } X' Y' \wedge X' \succ \text{Pareto } \mathcal{X}$

**using** *pareto-optimal-def* **by** *blast*

**thus** *?thesis*

**by** *auto*

qed

Using monotonicity instead of local non-satiation proves the First Welfare Theorem.

**corollary** *first-welfare-thm-monotone:*

**assumes**  $\forall M \in \text{carrier}. (\forall x > M. x \in \text{carrier})$

**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{monotone-preference consumption-set } Pr[i]$   
**and**  $Price > 0$

**assumes** *competitive-equilibrium*  $Price \mathcal{X} \mathcal{Y}$

**shows** *pareto-optimal*  $\mathcal{X} \mathcal{Y}$

**using** *all-larger-zero-in-csset assms(2) assms(3) assms(4)*

*first-welfare-theorem-priv-own unbounded-above-mono-imp-lms* **by** *blast*

end

end

end

## 9 Arrow-Debreu model

**theory** *Arrow-Debreu-Model*

**imports**

*../Preferences*

*../Preferences*

*../Utility-Functions*

*../Argmax*

*Consumers*

*Common*

**begin**

**locale** *pre-arrow-debreu-model* =

**fixes** *production-sets* ::  $'f \Rightarrow ('a::\text{ordered-euclidean-space}) \text{ set}$

**fixes** *consumption-set* :: 'a set  
**fixes** *agents* :: 'i set  
**fixes** *firms* :: 'f set  
**fixes**  $\mathcal{E}$  :: 'i  $\Rightarrow$  'a ( $\mathcal{E}[-]$ )  
**fixes** *Pref* :: 'i  $\Rightarrow$  'a relation (*Pr*[-])  
**fixes** *U* :: 'i  $\Rightarrow$  'a  $\Rightarrow$  real (*U*[-])  
**fixes**  $\Theta$  :: 'i  $\Rightarrow$  'f  $\Rightarrow$  real ( $\Theta[-,-]$ )  
**assumes** *cons-set-props*: *arrow-debreu-consum-set consumption-set*  
**assumes** *agent-props*:  $i \in \text{agents} \Rightarrow \text{eucl-ordinal-utility consumption-set } (Pr[i])$   
(*U*[*i*])  
**assumes** *firms-comp-owned*:  $j \in \text{firms} \Rightarrow (\sum_{i \in \text{agents}} \Theta[i,j]) = 1$   
**assumes** *finite-nonepty-agents*: *finite agents* **and**  $\text{agents} \neq \{\}$

**sublocale** *pre-arrow-debreu-model*  $\subseteq$  *pareto-ordering agents U*  
.

**context** *pre-arrow-debreu-model*  
**begin**

Calculate wealth of individual i in context of Private Ownership economy.

**context**  
**begin**

**private abbreviation** *poe-wealth*  
**where**  
*poe-wealth*  $P \ i \ Y \equiv P \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms}} \Theta[i,j] *_{\mathbb{R}} (P \cdot Y \ j))$

## 9.1 Feasibility

**private abbreviation** *feasible*  
**where**  
*feasible*  $X \ Y \equiv \text{feasible-private-ownership agents firms } \mathcal{E} \text{ consumption-set production-sets } X \ Y$

**private abbreviation** *calculate-value*  
**where**  
*calculate-value*  $P \ x \equiv P \cdot x$

## 9.2 Profit maximisation

In a production economy (which this is) we need to specify profit maximisation.

**definition** *profit-maximisation*  
**where**  
*profit-maximisation*  $P \ S = \text{arg-max-set } (\lambda x. P \cdot x) \ S$

### 9.3 Competitive Equilibrium

Competitive equilibrium in context of production economy with private ownership. This includes the profit maximisation condition.

**definition** *competitive-equilibrium*

**where**

*competitive-equilibrium*  $P X Y \iff \text{feasible } X Y \wedge$   
 $(\forall j \in \text{firms. } (Y j) \in \text{profit-maximisation } P (\text{production-sets } j)) \wedge$   
 $(\forall i \in \text{agents. } (X i) \in \text{arg-max-set } U[i] (\text{budget-constraint } (\text{calculate-value } P)$   
 $\text{consumption-set } (\text{poe-wealth } P i Y)))$

**lemma** *competitive-equilibriumD* [dest]:

**assumes** *competitive-equilibrium*  $P X Y$

**shows** *feasible*  $X Y \wedge$

$(\forall j \in \text{firms. } (Y j) \in \text{profit-maximisation } P (\text{production-sets } j)) \wedge$

$(\forall i \in \text{agents. } (X i) \in \text{arg-max-set } U[i] (\text{budget-constraint } (\text{calculate-value } P)$   
 $\text{consumption-set } (\text{poe-wealth } P i Y)))$

**using** *assms* **by** (*simp add: competitive-equilibrium-def*)

**lemma** *compet-max-profit*:

**assumes**  $j \in \text{firms}$

**assumes** *competitive-equilibrium*  $P X Y$

**shows**  $Y j \in \text{profit-maximisation } P (\text{production-sets } j)$

**using** *assms(1)* *assms(2)* **by** *blast*

### 9.4 Pareto Optimality

**definition** *pareto-optimal*

**where**

*pareto-optimal*  $X Y \iff$   
 $(\text{feasible } X Y \wedge$   
 $(\nexists X' Y'. \text{feasible } X' Y' \wedge X' \succ \text{Pareto } X))$

**lemma** *pareto-optimalI*[intro]:

**assumes** *feasible*  $X Y$

**and**  $\nexists X' Y'. \text{feasible } X' Y' \wedge X' \succ \text{Pareto } X$

**shows** *pareto-optimal*  $X Y$

**using** *pareto-optimal-def* *assms(1)* *assms(2)* **by** *blast*

**lemma** *pareto-optimalD*[dest]:

**assumes** *pareto-optimal*  $X Y$

**shows** *feasible*  $X Y$  **and**  $\nexists X' Y'. \text{feasible } X' Y' \wedge X' \succ \text{Pareto } X$

**using** *pareto-optimal-def* *assms* **by** *auto*

**lemma** *util-fun-def-holds*:

**assumes**  $i \in \text{agents}$

**and**  $x \in \text{consumption-set}$

**and**  $y \in \text{consumption-set}$

**shows**  $x \succeq [\text{Pr}[i]] y \iff U[i] x \geq U[i] y$

**proof**  
**assume**  $x \succeq_{[Pr[i]]} y$   
**show**  $U[i] x \geq U[i] y$   
**by** (*meson*  $\langle x \succeq_{[Pr[i]]} y \rangle$  *agent-props assms eucl-ordinal-utility-def ordinal-utility-def*)  
**next**  
**assume**  $U[i] x \geq U[i] y$   
**have** *eucl-ordinal-utility consumption-set* ( $Pr[i]$ ) ( $U[i]$ )  
**by** (*simp add: agent-props assms*)  
**then show**  $x \succeq_{[Pr[i]]} y$   
**by** (*meson*  $\langle U[i] y \leq U[i] x \rangle$  *assms(2) assms(3) eucl-ordinal-utility-def ordinal-utility.util-def-conf*)  
**qed**

**lemma** *base-pref-is-ord-eucl-rpr*:  $i \in \text{agents} \implies \text{rational-preference consumption-set } Pr[i]$   
**using** *agent-props ord-eucl-utility-imp-rpr real-vector-rpr.have-rpr* **by** *blast*

**lemma** *prof-max-ge-all-in-pset*:  
**assumes**  $j \in \text{firms}$   
**assumes**  $Y j \in \text{profit-maximisation } P$  (*production-sets*  $j$ )  
**shows**  $\forall y \in \text{production-sets } j. P \cdot Y j \geq P \cdot y$   
**using** *all-leq assms(2) profit-maximisation-def* **by** *fastforce*

## 9.5 Lemmas for final result

Strictly preferred bundles are strictly more expensive.

**lemma** *all-prefered-are-more-expensive*:  
**assumes**  $i \in \text{agents}$   
**assumes** *equil: competitive-equilibrium*  $P \mathcal{X} \mathcal{Y}$   
**assumes**  $z \in \text{consumption-set}$   
**assumes**  $(U i) z > (U i) (\mathcal{X} i)$   
**shows**  $z \cdot P > P \cdot (\mathcal{X} i)$   
**proof** (*rule ccontr*)  
**assume** *neg-as* :  $\neg(z \cdot P > P \cdot (\mathcal{X} i))$   
**have** *xp-leq* :  $z \cdot P \leq P \cdot (\mathcal{X} i)$   
**using**  $\langle \neg z \cdot P > P \cdot (\mathcal{X} i) \rangle$  **by** *auto*  
**have** *x-in-argmax*:  $(\mathcal{X} i) \in \text{arg-max-set } U[i]$  (*budget-constraint* (*calculate-value*  $P$ ) *consumption-set* (*poe-wealth*  $P i \mathcal{Y}$ ))  
**using** *equil i-agt* **by** *blast*  
**hence** *x-in*:  $\mathcal{X} i \in (\text{budget-constraint} (\text{calculate-value } P) \text{consumption-set} (\text{poe-wealth } P i \mathcal{Y}))$   
**using** *argmax-sol-in-s* [*of*  $(\mathcal{X} i) U[i]$  *budget-constraint* (*calculate-value*  $P$ ) *consumption-set* (*poe-wealth*  $P i \mathcal{Y}$ )]  
**by** *blast*  
**hence** *z-in-budget*:  $z \in (\text{budget-constraint} (\text{calculate-value } P) \text{consumption-set} (\text{poe-wealth } P i \mathcal{Y}))$   
**proof** –  
**have** *z-leq-endow*:  $P \cdot z \leq P \cdot (\mathcal{X} i)$   
**by** (*metis xp-leq inner-commute*)

```

have z-in-cons:  $z \in \text{consumption-set}$ 
using assms by auto
then show ?thesis
using x-in budget-constraint-def z-leq-endow
proof –
have  $\forall r. P \cdot \mathcal{X} \ i \leq r \longrightarrow P \cdot z \leq r$ 
using z-leq-endow by linarith
then show ?thesis
using budget-constraint-def x-in z-in-cons
by (simp add: budget-constraint-def)
qed
qed
have nex-prop:  $\nexists e. e \in (\text{budget-constraint} (\text{calculate-value } P) \text{consumption-set} (\text{poe-wealth } P \ i \ \mathcal{Y})) \wedge U[i] \ e > U[i] (\mathcal{X} \ i)$ 
using no-better-in-s[of X i U[i] budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y)]
x-in-argmax by blast
have  $z \in \text{budget-constraint} (\text{calculate-value } P) \text{consumption-set} (\text{poe-wealth } P \ i \ \mathcal{Y}) \wedge U[i] \ z > U[i] (\mathcal{X} \ i)$ 
using assms z-in-budget by blast
thus False using nex-prop
by blast
qed

```

Given local non-satiation, argmax will use the entire budget.

**lemma** *am-utilises-entire-bgt*:

```

assumes i-agts:  $i \in \text{agents}$ 
assumes lms : local-nonsatiation consumption-set Pr[i]
assumes argmax-sol :  $X \in \text{arg-max-set } U[i] (\text{budget-constraint} (\text{calculate-value } P) \text{consumption-set} (\text{poe-wealth } P \ i \ Y))$ 
shows  $P \cdot X = P \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms}} \Theta[i,j] *_{\mathbb{R}} (P \cdot Y \ j))$ 
proof –
let ?wlt =  $P \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms}} \Theta[i,j] *_{\mathbb{R}} (P \cdot Y \ j))$ 
let ?bc = budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y)
have xin:  $X \in \text{budget-constraint} (\text{calculate-value } P) \text{consumption-set} (\text{poe-wealth } P \ i \ Y)$ 
using argmax-sol-in-s [of X U[i] ?bc] argmax-sol by blast
hence is-leq:  $X \cdot P \leq (\text{poe-wealth } P \ i \ Y)$ 
by (metis (mono-tags, lifting) budget-constraint-def inner-commute mem-Collect-eq)
have not-less:  $\neg X \cdot P < (\text{poe-wealth } P \ i \ Y)$ 
proof
assume neg:  $X \cdot P < (\text{poe-wealth } P \ i \ Y)$ 
have bgt-leq:  $\forall x \in ?bc. U[i] \ X \geq U[i] \ x$ 
using leq-all-in-sol [of X U[i] ?bc]
all-leq [of X U[i] ?bc]
argmax-sol by blast

```



```

define s-low where
  s-low = {x . P · x < ?wlt}
have  $\exists e > 0. \text{ball } X e \subseteq \textit{s-low}$ 
proof –
  have x-in-budget: P · X < ?wlt
    by (metis inner-commute neg)
  have s-low-open: open s-low
    using open-halfspace-lt s-low-def by blast
  then show ?thesis
    using s-low-open open-contains-ball-eq
      s-low-def x-in-budget by blast
qed
obtain e where
  e > 0 and e: ball X e  $\subseteq$  s-low
  using  $\langle \exists e > 0. \text{ball } X e \subseteq \textit{s-low} \rangle$  by blast
obtain y where
  y-props: y  $\in$  ball X e y  $\succ$ [Pref i] X
  using  $\langle 0 < e \rangle$  xin assms(2) budget-constraint-def
  by (metis (no-types, lifting) lns-alt-def2 mem-Collect-eq)
  have y  $\in$  budget-constraint (calculate-value P) consumption-set (poe-wealth P
i Y)
proof –
  have y  $\in$  s-low
    using  $\langle y \in \text{ball } X e \rangle$  e by blast
  moreover have y  $\in$  consumption-set
    by (meson agent-props eucl-ordinal-utility-def i-agts ordinal-utility-def
y-props(2))
  moreover have P · y  $\leq$  poe-wealth P i Y
    using calculation(1) s-low-def by auto
  ultimately show ?thesis
    by (simp add: budget-constraint-def)
qed
then show False
  using bgt-leq i-agts y-props(2) util-fun-def-holds xin budget-constraint-def
  by (metis (no-types, lifting) mem-Collect-eq)
qed
then show ?thesis
  by (metis inner-commute is-leq
    less-eq-real-def)
qed

corollary x-equil-x-ext-budget:
  assumes i-agt: i  $\in$  agents
  assumes lns : local-nonsatiation consumption-set Pr[i]
  assumes equilibrium : competitive-equilibrium P X Y
  shows P · X i = P ·  $\mathcal{E}[i]$  +  $(\sum_{j \in \text{firms.}} \Theta[i,j] *_R (P \cdot Y j))$ 
proof –
  have X i  $\in$  arg-max-set U[i] (budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y))

```

**using** *equilibrium i-agt by blast*  
**then show** *?thesis*  
**using** *am-utilises-entire-bgt i-agt lns by blast*  
**qed**

**lemma** *same-price-in-argmax* :  
**assumes** *i-agt: i ∈ agents*  
**assumes** *lns : local-nonsatiation consumption-set Pr[i]*  
**assumes** *x ∈ arg-max-set (U[i]) (budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y))*  
**assumes** *y ∈ arg-max-set (U[i]) (budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y))*  
**shows**  $(P \cdot x) = (P \cdot y)$   
**using** *am-utilises-entire-bgt assms lns*  
**by** *(metis (no-types) am-utilises-entire-bgt assms(3) assms(4) i-agt lns)*

Greater or equal utility implies greater or equal value.

**lemma** *utility-ge-price-ge* :  
**assumes** *agts: i ∈ agents*  
**assumes** *lns : local-nonsatiation consumption-set Pr[i]*  
**assumes** *equil: competitive-equilibrium P X Y*  
**assumes** *geq: U[i] z ≥ U[i] (X i)*  
**and** *z ∈ consumption-set*  
**shows**  $P \cdot z \geq P \cdot (X i)$   
**proof** –  
**let** *?bc = (budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y))*  
**have** *not-in : z ∉ arg-max-set (U[i]) ?bc ⇒*  
 $P \cdot z > (P \cdot \mathcal{E}[i]) + (\sum_{j \in (\text{firms})}. (\Theta[i, j] *_{\mathbb{R}} (P \cdot Y j)))$   
**proof** –  
**assume** *z ∉ arg-max-set (U[i]) ?bc*  
**moreover have** *X i ∈ arg-max-set (U[i]) ?bc*  
**using** *competitive-equilibriumD assms pareto-optimal-def*  
**by** *auto*  
**ultimately have** *z ∉ budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y)*  
**by** *(meson geq leq-all-in-sol)*  
**then show** *?thesis*  
**using** *budget-constraint-def assms*  
**by** *(simp add: budget-constraint-def)*  
**qed**  
**have** *x-in-argmax: (X i) ∈ arg-max-set U[i] ?bc*  
**using** *agts equil by blast*  
**hence** *x-in-budget: (X i) ∈ ?bc*  
**using** *argmax-sol-in-s [of (X i) U[i] ?bc] by blast*  
**have**  $U[i] z = U[i] (X i) \implies P \cdot z \geq P \cdot (X i)$   
**proof** *(rule contrapos-pp)*  
**assume** *con-neg: ¬ P · z ≥ P · (X i)*  
**then have**  $P \cdot z < P \cdot (X i)$

```

    by linarith
  then have z-in-argmax:  $z \in \text{arg-max-set } U[i]$  ?bc
  proof -
    have  $P \cdot (X\ i) = P \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms.}} \Theta[i,j] *_R (P \cdot Y\ j))$ 
      using agts am-utilises-entire-bgt lns x-in-argmax by blast
    then show ?thesis
      by (metis (no-types) con-neg less-eq-real-def not-in)
    qed
  have z-budget-utilisation:  $P \cdot z = P \cdot (X\ i)$ 
    by (metis (no-types) agts am-utilises-entire-bgt lns x-in-argmax z-in-argmax)
  have  $P \cdot (X\ i) = P \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms.}} \Theta[i,j] *_R (P \cdot Y\ j))$ 
    using agts am-utilises-entire-bgt lns x-in-argmax by blast
  show  $\neg U[i]\ z = U[i]\ (X\ i)$ 
    using z-budget-utilisation con-neg by linarith
  qed
  thus ?thesis
    by (metis (no-types) agts am-utilises-entire-bgt eq-iff eucl-less-le-not-le lns not-in x-in-argmax)
  qed

```

**lemma** *commutativity-sums-over-funs*:

```

  fixes  $X :: 'x\ \text{set}$ 
  fixes  $Y :: 'y\ \text{set}$ 
  shows  $(\sum_{i \in X.} \sum_{j \in Y.} (f\ i\ j *_R C \cdot g\ j)) = (\sum_{j \in Y.} \sum_{i \in X.} (f\ i\ j *_R C \cdot g\ j))$ 
  using Groups-Big.comm-monoid-add-class.sum.swap by auto

```

**lemma** *assoc-fun-over-sum*:

```

  fixes  $X :: 'x\ \text{set}$ 
  fixes  $Y :: 'y\ \text{set}$ 
  shows  $(\sum_{j \in Y.} \sum_{i \in X.} f\ i\ j *_R C \cdot g\ j) = (\sum_{j \in Y.} (\sum_{i \in X.} f\ i\ j) *_R C \cdot g\ j)$ 
  by (simp add: inner-sum-left scaleR-left.sum)

```

Walras' law in context of production economy with private ownership. That is, in an equilibrium demand equals supply.

**lemma** *walras-law*:

```

  assumes  $\bigwedge i. i \in \text{agents} \implies \text{local-nonsatiation consumption-set } Pr[i]$ 
  assumes  $(\forall i \in \text{agents. } (X\ i) \in \text{arg-max-set } U[i])$  (budget-constraint (calculate-value P) consumption-set (poe-wealth P i Y))
  shows  $P \cdot (\sum_{i \in \text{agents.}} (X\ i)) = P \cdot ((\sum_{i \in \text{agents.}} \mathcal{E}[i]) + (\sum_{j \in \text{firms.}} Y\ j))$ 
  proof -
    have value-equal:  $P \cdot (\sum_{i \in \text{agents.}} (X\ i)) = P \cdot (\sum_{i \in \text{agents.}} \mathcal{E}[i]) + (\sum_{i \in \text{agents.}} \sum_{f \in \text{firms.}} \Theta[i,f] *_R (P \cdot Y\ f))$ 
      proof -
        have all-exhaust-bgt:  $\forall i \in \text{agents. } P \cdot (X\ i) = P \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms.}} \Theta[i,j] *_R (P \cdot (Y\ j)))$ 
          using assms am-utilises-entire-bgt by blast
        then show ?thesis
          by (simp add: all-exhaust-bgt inner-sum-right sum.distrib)
      qed
    qed
  qed

```

**have** eq-1:  $(\sum_{i \in \text{agents}}. \sum_{j \in \text{firms}}. (\Theta[i,j] *_R P \cdot Y j)) = (\sum_{j \in \text{firms}}. \sum_{i \in \text{agents}}. (\Theta[i,j] *_R P \cdot Y j))$   
**using** commutativity-sums-over-funs [of  $\Theta P Y$  firms agents] **by** blast  
**hence** eq-2:  $P \cdot (\sum_{i \in \text{agents}}. X i) = P \cdot (\sum_{i \in \text{agents}}. \mathcal{E}[i]) + (\sum_{j \in \text{firms}}. \sum_{i \in \text{agents}}. \Theta[i,j] *_R P \cdot Y j)$   
**using** value-equal **by** auto  
**also have** eq-3:  $\dots = P \cdot (\sum_{i \in \text{agents}}. \mathcal{E}[i]) + (\sum_{j \in \text{firms}}. (\sum_{i \in \text{agents}}. \Theta[i,j] *_R P \cdot Y j))$   
**using** assoc-fun-over-sum [of  $\Theta P Y$  agents firms] **by** auto  
**also have** eq-4:  $\dots = P \cdot (\sum_{i \in \text{agents}}. \mathcal{E}[i]) + (\sum_{f \in \text{firms}}. P \cdot Y f)$   
**using** firms-comp-owned **by** auto  
**have** comp-wise-inner:  $P \cdot (\sum_{i \in \text{agents}}. X i) - (P \cdot (\sum_{i \in \text{agents}}. \mathcal{E}[i]) - (\sum_{f \in \text{firms}}. P \cdot Y f)) = 0$   
**using** eq-1 eq-2 eq-3 eq-4 **by** linarith  
**then show** ?thesis  
**by** (simp add: inner-right-distrib inner-sum-right)  
**qed**

**lemma** walras-law-in-compeq:

**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{local-nonsatiation consumption-set } Pr[i]$   
**assumes** competitive-equilibrium  $P X Y$   
**shows**  $P \cdot ((\sum_{i \in \text{agents}}. (X i)) - (\sum_{i \in \text{agents}}. \mathcal{E}[i]) - (\sum_{j \in \text{firms}}. Y j)) = 0$   
**proof** –  
**have**  $P \cdot (\sum_{i \in \text{agents}}. (X i)) = P \cdot ((\sum_{i \in \text{agents}}. \mathcal{E}[i]) + (\sum_{j \in \text{firms}}. Y j))$   
**using** assms(1) assms(2) walras-law **by** auto  
**then show** ?thesis  
**by** (simp add: inner-diff-right inner-right-distrib)  
**qed**

## 9.6 First Welfare Theorem

Proof of First Welfare Theorem in context of production economy with private ownership.

**theorem** first-welfare-theorem-priv-own:

**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{local-nonsatiation consumption-set } Pr[i]$   
**and**  $Price > 0$   
**assumes** competitive-equilibrium  $Price \mathcal{X} \mathcal{Y}$   
**shows** pareto-optimal  $\mathcal{X} \mathcal{Y}$

**proof** (rule ccontr)

**assume** neg-as:  $\neg \text{pareto-optimal } \mathcal{X} \mathcal{Y}$   
**have** equili-feasible : feasible  $\mathcal{X} \mathcal{Y}$   
**using** assms **by** (simp add: competitive-equilibrium-def)  
**obtain**  $X' Y'$  **where**  
*xprime-pareto*: feasible  $X' Y' \wedge$   
 $(\forall i \in \text{agents}. U[i] (X' i) \geq U[i] (X i)) \wedge$   
 $(\exists i \in \text{agents}. U[i] (X' i) > U[i] (X i))$   
**using** equili-feasible pareto-optimal-def  
*pareto-dominating-def neg-as* **by** auto  
**have** is-feasible: feasible  $X' Y'$

**using** *xprime-pareto* **by** *blast*  
**have** *xprime-leq-y*:  $\forall i \in \text{agents}. (\text{Price} \cdot (X' i) \geq$   
 $(\text{Price} \cdot \mathcal{E}[i]) + (\sum_{j \in \text{firms}}. \Theta[i,j] *_{\mathcal{R}} (\text{Price} \cdot \mathcal{Y} j)))$   
**proof**  
**fix** *i*  
**assume** *as*:  $i \in \text{agents}$   
**have** *xprime-cons*:  $X' i \in \text{consumption-set}$   
**using** *feasible-private-ownershipD* **as** *is-feasible* **by** *blast*  
**have** *x-leq-xprime*:  $U[i] (X' i) \geq U[i] (\mathcal{X} i)$   
**using**  $\langle i \in \text{agents} \rangle$  *xprime-pareto* **by** *blast*  
**have** *lms-pref*: *local-nonsatiation consumption-set*  $Pr[i]$   
**using** *as* **assms** **by** *blast*  
**hence** *xprime-ge-x*:  $\text{Price} \cdot (X' i) \geq \text{Price} \cdot (\mathcal{X} i)$   
**using** *x-leq-xprime* *xprime-cons* **as** *assms* *utility-ge-price-ge* **by** *blast*  
**then show**  $\text{Price} \cdot (X' i) \geq (\text{Price} \cdot \mathcal{E}[i]) + (\sum_{j \in \text{firms}}. \Theta[i,j] *_{\mathcal{R}} (\text{Price} \cdot$   
 $\mathcal{Y} j))$   
**using** *xprime-ge-x*  $\langle i \in \text{agents} \rangle$  *lms-pref* **assms** *x-equil-x-ext-budget* **by** *fastforce*  
**qed**  
**have** *ex-greater-value* :  $\exists i \in \text{agents}. \text{Price} \cdot (X' i) > \text{Price} \cdot (\mathcal{X} i)$   
**proof**(*rule ccontr*)  
**assume** *cpos* :  $\neg(\exists i \in \text{agents}. \text{Price} \cdot (X' i) > \text{Price} \cdot (\mathcal{X} i))$   
**obtain** *i* **where**  
*obt-witness* :  $i \in \text{agents} (U[i] (X' i) > U[i] (\mathcal{X} i))$   
**using** *xprime-pareto* **by** *blast*  
**show** *False*  
**by** (*metis all-prefered-are-more-expensive* *assms*(3) *cpos*  
*feasible-private-ownershipD*(2) *inner-commute* *xprime-pareto*)  
**qed**  
**have** *dom-g* :  $\text{Price} \cdot (\sum_{i \in \text{agents}}. X' i) > \text{Price} \cdot (\sum_{i \in \text{agents}}. (\mathcal{X} i))$  (**is** - >  
- . ?*x-sum*)  
**proof**-  
**have**  $(\sum_{i \in \text{agents}}. \text{Price} \cdot X' i) > (\sum_{i \in \text{agents}}. \text{Price} \cdot (\mathcal{X} i))$   
**by** (*metis (mono-tags, lifting)* *xprime-leq-y* *assms*(1,3) *ex-greater-value*  
*finite-nonepty-agents* *sum-strict-mono-ex1* *x-equil-x-ext-budget*)  
**thus**  $\text{Price} \cdot (\sum_{i \in \text{agents}}. X' i) > \text{Price} \cdot ?x\text{-sum}$   
**by** (*simp* *add: inner-sum-right*)  
**qed**  
**let** ?*y-sum* =  $(\sum_{j \in \text{firms}}. \mathcal{Y} j)$   
**have** *equili-walras-law*:  $\text{Price} \cdot ?x\text{-sum} =$   
 $(\sum_{i \in \text{agents}}. \text{Price} \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms}}. \Theta[i,j] *_{\mathcal{R}} (\text{Price} \cdot \mathcal{Y} j)))$  (**is** - = ?*ws*)  
**proof**-  
**have**  $\forall i \in \text{agents}. \text{Price} \cdot \mathcal{X} i = \text{Price} \cdot \mathcal{E}[i] + (\sum_{j \in \text{firms}}. \Theta[i,j] *_{\mathcal{R}} (\text{Price} \cdot \mathcal{Y}$   
 $j))$   
**by** (*metis (no-types, lifting)* *assms*(1,3) *x-equil-x-ext-budget*)  
**then show** ?*thesis*  
**by** (*simp* *add: inner-sum-right*)  
**qed**  
**also have** *remove-firm-pct*:  $\dots = \text{Price} \cdot (\sum_{i \in \text{agents}}. \mathcal{E}[i]) + (\text{Price} \cdot ?y\text{-sum})$   
**proof**-

**have** *equals-inner-price:0* =  $Price \cdot (?x\text{-sum} - ((\sum_{i \in \text{agents}} \mathcal{E} i) + ?y\text{-sum}))$   
**by** (*metis* (*no-types*) *diff-diff-add* *assms(1,3)* *walras-law-in-compeq*)  
**have**  $Price \cdot ?x\text{-sum} = Price \cdot ((\sum_{i \in \text{agents}} \mathcal{E} i) + ?y\text{-sum})$   
**by** (*metis* (*no-types*) *equals-inner-price* *inner-diff-right* *right-minus-eq*)  
**then show** *?thesis*  
**by** (*simp* *add: equili-walras-law* *inner-right-distrib*)  
**qed**  
**have** *xp-l-yp*:  $(\sum_{i \in \text{agents}} X' i) \leq (\sum_{i \in \text{agents}} \mathcal{E}[i]) + (\sum_{f \in \text{firms}} Y' f)$   
**using** *feasible-private-ownership-def* *is-feasible* **by** *blast*  
**hence** *yprime-sgr-y*:  $Price \cdot (\sum_{i \in \text{agents}} \mathcal{E}[i]) + Price \cdot (\sum_{f \in \text{firms}} Y' f) >$   
*?ws*  
**proof** –  
**have**  $Price \cdot (\sum_{i \in \text{agents}} X' i) \leq Price \cdot ((\sum_{i \in \text{agents}} \mathcal{E}[i]) + (\sum_{j \in \text{firms}} Y' j))$   
*Y' j*)  
**by** (*metis* *xp-l-yp* *atLeastAtMost-iff* *inner-commute*  
*interval-inner-leI(2)* *less-imp-le* *order-refl* *assms(2)*)  
**hence** *?ws* <  $Price \cdot ((\sum_{i \in \text{agents}} \mathcal{E} i) + (\sum_{j \in \text{firms}} Y' j))$   
**using** *dom-g* *equili-walras-law* **by** *linarith*  
**then show** *?thesis*  
**by** (*simp* *add: inner-right-distrib*)  
**qed**  
**have** *Y-is-optimum*:  $\forall j \in \text{firms}. \forall y \in \text{production-sets } j. Price \cdot \mathcal{Y} j \geq Price \cdot y$   
**using** *assms* *prof-max-ge-all-in-pset* **by** *blast*  
**have** *yprime-in-prod-set*:  $\forall j \in \text{firms}. Y' j \in \text{production-sets } j$   
**using** *feasible-private-ownershipD* *xprime-pareto* **by** *fastforce*  
**hence**  $\forall j \in \text{firms}. \forall y \in \text{production-sets } j. Price \cdot \mathcal{Y} j \geq Price \cdot y$   
**using** *Y-is-optimum* **by** *blast*  
**hence** *Y-ge-yprime*:  $\forall j \in \text{firms}. Price \cdot \mathcal{Y} j \geq Price \cdot Y' j$   
**using** *yprime-in-prod-set* **by** *blast*  
**hence** *yprime-p-leq-Y*:  $Price \cdot (\sum_{f \in \text{firms}} Y' f) \leq Price \cdot ?y\text{-sum}$   
**by** (*simp* *add: Y-ge-yprime* *inner-sum-right* *sum-mono*)  
**then show** *False*  
**using** *remove-firm-pct* *yprime-sgr-y* **by** *linarith*  
**qed**

Equilibrium cannot be Pareto dominated.

**lemma** *equilibria-dom-eachother*:

**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{local-nonsatiation consumption-set } Pr[i]$   
**and**  $Price > 0$

**assumes** *equil: competitive-equilibrium*  $Price \mathcal{X} \mathcal{Y}$

**shows**  $\nexists X' Y'. \text{competitive-equilibrium } P X' Y' \wedge X' \succ \text{Pareto } \mathcal{X}$

**proof** –

**have** *pareto-optimal*  $\mathcal{X} \mathcal{Y}$

**by** (*meson* *equil* *first-welfare-theorem-priv-own* *assms*)

**hence**  $\nexists X' Y'. \text{feasible } X' Y' \wedge X' \succ \text{Pareto } \mathcal{X}$

**using** *pareto-optimal-def* **by** *blast*

**thus** *?thesis*

**by** *auto*

**qed**

Using monotonicity instead of local non-satiation proves the First Welfare Theorem.

**corollary** *first-welfare-thm-monotone:*

**assumes**  $\forall M \in \text{carrier}. (\forall x > M. x \in \text{carrier})$

**assumes**  $\bigwedge i. i \in \text{agents} \implies \text{monotone-preference consumption-set } Pr[i]$

**and**  $Price > 0$

**assumes** *competitive-equilibrium Price  $\mathcal{X}$   $\mathcal{Y}$*

**shows** *pareto-optimal  $\mathcal{X}$   $\mathcal{Y}$*

**by** (*meson arrow-debreu-consum-set-def assms cons-set-props first-welfare-theorem-priv-own unbounded-above-mono-imp-lns*)

**end**

**end**

**end**

## 10 Related work

[2]

## References

- [1] K. J. Arrow, A. Sen, and K. Suzumura. *Handbook of Social Choice and Welfare*, volume 2. Elsevier, 2010.
- [2] S. Tadelis. *Game Theory: An Introduction*. Princeton University Press, 2013.